ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - KI141502

PENGENALAN WAJAH MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK

VINCENT DANIEL WIN
NRP 05111440000139

Dosen Pembimbing I
Dr. Eng. Nanik Suciati, S.Kom., M.Kom.

Dosen Pembimbing II
Anny Yuniarti, S.Kom., M.Comp.Sc.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KI141502

PENGENALAN WAJAH MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK

VINCENT DANIEL WIN
NRP 05111440000139

Dosen Pembimbing I
Dr. Eng. Nanik Suciati, S.Kom., M.Kom.

Dosen Pembimbing II
Anny Yuniarti, S.Kom., M.Comp.Sc.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

UNDERGRADUATE THESES - KI141502

FACE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK

Vincent Daniel Win
NRP 05111440000139

Supervisor I
Dr. Eng. Nanik Suciati, S.Kom., M.Kom.

Supervisor II
Anny Yuniarti, S.Kom., M.Comp.Sc.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND COMMUNICATION
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2018

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

PENGENALAN WAJAH MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Cerdas Visual
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh
VINCENT DANIEL WIN
NRP: 05111440000139

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1 Dr. Eng. Nanik Suciati, S.Kom., M.Kom. .....
NIP: 19710428 199412 2 001 (Pembimbing 1)

2 Anny Yuniarti, S.Kom., M.Comp.Sc. .....
NIP: 19810622 200501 2 002 (Pembimbing 2)

SURABAYA
JUNI, 2018

[Halaman ini sengaja dikosongkan]

PENGENALAN WAJAH MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK

Nama Mahasiswa	**: VINCENT DANIEL WIN**
NRP	**: 05111440000139**
Jurusan	**: Teknik Informatika FTIF-ITS**
Dosen Pembimbing 1	**: Dr. Eng. Nanik Suciati, S.Kom., M.Kom.**
Dosen Pembimbing 2	**: Anny Yuniarti, S.Kom., M. Comp. Sc.**

ABSTRAK

Beberapa tahun belakangan metode deep learning menarik perhatian karena menghasilkan kinerja yang sangat baik pada beragam aplikasi. Perkembangan yang pesat ini disebabkan oleh meledaknya jumlah data dan kemampuan proses komputer.

Tugas akhir ini mengimplementasikan metode Deep Learning Convolutional Neural Network (CNN) pada klasifikasi data wajah. CNN sendiri sudah terbukti dapat mengklasifikasikan data gambar (2 dimensi spatial) dengan nilai akurasi yang tinggi, namun memerlukan dataset yang banyak. Dalam tugas akhir ini, diimplementasikan VGG16 yang merupakan sebuah arsitektur CNN, untuk pengenalan wajah pada dataset LFW dan faces94.

Dari uji coba menggunakan dataset faces94, ditemukan bahwa penggunaan fungsi aktivasi Leaky ReLU lebih baik dibandingkan ReLU dalam hal kecepatan mencapai konvergensi ketika training. Sedangkan nilai akurasi sistem dapat ditingkatkan hingga 99,6% pada dataset LFW, dengan cara melakukan augmentasi data dengan proses filter dan menerapkan face alignment.

Kata kunci: Deep Learning, Convolutional Neural Network, Pengenalan Wajah, Klasifikasi Wajah.

[Halaman ini sengaja dikosongkan]

FACE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK

Student's Name	**: VINCENT DANIEL WIN**
Student's ID	**: 05111440000139**
Department	**: Teknik Informatika FTIF-ITS**
First Advisor	**: Dr. Eng. Nanik Suciati, S.Kom., M.Kom.**
Second Advisor	**: Anny Yuniarti, S.Kom., M.Comp.Sc.**

ABSTRACT

In recent years, deep learning has gained attention because of its flexibility in many major applications. Its recent fast development mainly attributed to the exploding growth of data availability and recent improvement in GPU technologies which utilize vector computation quickly.

This final project implements Convolutional Neural Network (CNN), a Deep Learning method mainly used to classify images, on the dataset of face images. Using CNN yield high accuracy, but generally requires vast amount of data to be effective. In this final project, VGG16 which is a CNN architecture is implemented for face recognition using the LFW and faces94 dataset.

From the testing conducted, Leaky ReLU gives better performance which resulted in faster training convergence when compared to ReLU in face recognition using faces94 dataset. Meanwhile, accuracy can be improved up to 99.6% on LFW dataset by performing data augmentation using filtering and applying face alignment.

Keywords: Deep Learning, Convolutional Neural Network, Face Recognition, Face Classification.

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Puji dan syukur kepada Tuhan Allah Mahakudus yang telah melimpahkan berkat dan rahmat-Nya sehingga penulis dapat menyelesaikan penulisan Tugas Akhir yang berjudul "**PENGENALAN WAJAH MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK**".

Pengerjaan Tugas Akhir ini merupakan suatu kesempatan yang sangat baik bagi penulis. Dengan pengerjaan Tugas Akhir ini, penulis dapat belajar lebih banyak untuk memperdalam dan meningkatkan apa yang telah didapatkan penulis selama menempuh perkuliahan di Jurusan Teknik Informatika ITS. Dengan Tugas Akhir ini, penulis juga dapat menghasilkan suatu implementasi dari apa yang telah penulis pelajari.

Selesainya Tugas Akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak. Sehingga pada kesempatan ini penulis mengucapkan syukur dan terima kasih kepada keluarga, ayah dan ibu, yang telah senantiasa memberi dukungan dan menyediakan kecukupan bagi penulis dalam menapak jenjang pendidikan yang lebih tinggi dan menggapai cita-cita. Pada kesempatan ini, penulis juga berterima kasih kepada dosen pembimbing, Ibu Nanik dan Ibu Anny yang senantiasa membimbing dan memberi masukan dalam pengerjaan dan penyusunan tugas akhir dari awal hingga selesainya. Selain itu, penulis juga tak lupa berterima kasih untuk teman-teman yang selalu memberi dukungan dan saran selama proses implementasi tugas akhir, dan selama 4 tahun menempuh perkuliahan bersama-sama.

Semoga tugas akhir ini dapat bermanfaat bagi pembaca.

Surabaya, Juni 2018

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Manusia bergantung kepada pancaindra untuk mengenali dunia dan melakukan kegiatan sehari-hari, termasuk dalam bersosialisasi. Salah satu pancaindra ini adalah mata, yang digunakan dalam mengenali wajah dari orang-orang di sekitar kita. Mata manusia memiliki ketajaman dalam membedakan wajah, sebuah kemampuan yang diasah melalui pengalaman bersosialisasi bertahun-tahun. Dengan mengombinasikan cara kerja mata/otak, pengalaman, dan tenaga komputer, dapat dihasilkan sebuah sistem yang dapat mengenali wajah sebaik otak manusia. Berdasarkan gagasan ini, *Deep Learning* dikembangkan, sebuah pendekatan *machine learning* yang mengambil inspirasi dari cara kerja otak dalam menyelesaikan suatu masalah. Arsitektur ini termasuk mahal (membutuhkan banyak *resource*) dalam pengimplementasiannya karena menggunakan operasi transformasi non-linear yang berlapis-lapis, namun dengan peningkatan kekuatan proses yang eksponensial dan ledakan data selama 2 dekade belakangan, *Deep Learning* mulai dijelajahi dan dikembangkan lebih dalam.

Deep Learning merupakan salah satu bentuk pendekatan dalam *machine learning* dan pengembangan dari *Artifical Neural Network* (ANN), yang terinspirasi dari cara kerja otak dalam mempelajari dan menyelesaikan sebuah permasalahan. Metode ini sedang mengalami peningkatan popularitas baik di kalangan akademis dan ahli, maupun di kalangan umum. Sebuah gagasan bahwa komputer dapat menyelesaikan sebuah permasalahan, dengan cara *self-learning* memupuk perhatian publik terhadap model arsitektur ini.

Terlebih lagi pada tahun 2015, Google DeepMind berhasil mengembangkan sebuah AI yang dapat memainkan video game [1] dengan cara mempelajari video game tersebut piksel demi piksel. Setiap kali menemukan kegagalan, sistem akan mempelajari dan memperbaharui dirinya, berulang kali setiap generasi, hingga diperoleh hasil yang diinginkan yaitu skor tertinggi. Pendekatan ini secara teori memungkinkan *Deep Learning* menyelesaikan berbagai macam permasalahan berdasarkan pengalaman, tidak hanya terbatas pada kode yang ditanamkan pada sistemnya.

Pada tugas akhir ini, akan dicoba mengimplementasikan *Deep Learning* dalam bidang pengenalan wajah. Metode *Deep Learning* yang digunakan adalah *Convolutional Neural Network* (CNN). Dengan penggunaan metode ini, diharapkan diperoleh hasil dengan akurasi yang baik pada beberapa skenario uji coba yang akan diterapkan. Nilai akurasi minimal harapan adalah 90% pada dataset LFW dan faces94.

1.2 Rumusan Masalah

Beberapa rumusan masalah dalam tugas akhir ini adalah:

1. Bagaimana cara melakukan implementasi *Convolutional Neural Network* dalam pengenalan wajah?
2. Bagaimana mengetahui kinerja *Convolutional Neural Network* dalam pengenalan wajah?
3. Bagaimana cara meningkatkan kinerja *Convolutional Neural Network* dalam pengenalan wajah?

1.3 Batasan Masalah

Batasan masalah dalam melakukan implementasi adalah sebagai berikut:

1. Digunakan bahasa pemrograman *Python 3.x*.
2. Dataset yang digunakan adalah *Labeled Faces in the Wild* (LFW), berisi lebih dari 13.000 citra wajah yang telah diberi label [2] dan *faces94* yang berisi lebih dari 3.000 citra wajah [3].

1.4 Tujuan

Tujuan dari pengerjaan tugas akhir ini adalah mengimplementasikan *Convolutional Neural Network* untuk pengenalan wajah.

1.5 Manfaat

Tugas akhir ini memiliki manfaat sebagai berikut:

1. Dapat diaplikasikan dalam bidang sekuritas, menghasilkan sistem untuk melakukan verifikasi kepemilikan sebuah properti.
2. Dapat diaplikasikan dalam absensi/presensi mahasiswa dalam kegiatan perkuliahan, untuk mencegah "titip absen" yang sering terjadi di kalangan mahasiswa.
3. Dapat diaplikasikan dalam bidang sosial media, untuk mengenali wajah dalam sebuah gambar, dan melakukan *tagging* terhadap wajah.

1.6 Metodologi

a. Penyusunan proposal tugas akhir

Pada tahap ini akan dihasilkan proposal yang berisi latar belakang, permasalahan dan tujuan tugas akhir, serta manfaat dari tercapainya tujuan tugas akhir. Selain itu, dalam

proposal juga terdapat deskripsi umum, dan metode yang akan digunakan, serta tinjauan pustaka dan daftar referensi pendukung yang digunakan dalam menyusun tugas akhir. Jadwal kerja juga turut dicantumkan untuk transparansi proses pengerjaan tugas akhir dari awal hingga selesainya.

b. Studi literatur

Beberapa literatur yang diperlukan sebagai pendukung dalam mencapai tujuan tugas akhir adalah studi literatur mengenai *Convolutional Neural Network* dan kemampuannya dalam mengolah data citra, peninjauan dataset LFW yang akan digunakan dalam pelatihan *neural network*, dan metode preprocessing data yang diperlukan untuk mencapai hasil eksperimen yang optimal.

c. Analisis dan desain perangkat lunak

Tahap ini meliputi perancangan sistem pengenalan wajah CNN berdasarkan studi literatur dan pembelajaran konsep teknologi dari perangkat lunak yang ada. Tahap ini mendefinisikan alur dari implementasi. Langkah-langkah yang dikerjakan juga didefinisikan pada tahap ini.

d. Implementasi perangkat lunak

Implementasi merupakan tahap membangun rancangan program yang telah dibuat. Pada tahapan ini direalisasikan apa yang terdapat pada tahapan sebelumnya, sehingga menjadi sebuah program yang sesuai dengan apa yang telah direncanakan.

e. Pengujian dan evaluasi

Pada tahapan ini dilakukan uji coba pada data yang telah dikumpulkan. Pengujian dan evaluasi akan dilakukan dengan menggunakan bahasa pemrograman yang telah ditentukan. Tahapan ini dimaksudkan untuk mengevaluasi kesesuaian data dan program serta mencari masalah yang mungkin timbul dan mengadakan perbaikan jika terdapat kesalahan.

f. Penyusunan Buku Tugas Akhir

Pada tahapan ini disusun buku yang memuat dokumentasi mengenai pembuatan serta hasil dari implementasi perangkat lunak yang telah dibuat.

1.7 Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini:

BAB I PENDAHULUAN

Bab ini berisi latar belakang masalah, rumusan dan batasan masalah, tujuan dan manfaat pembuatan tugas akhir, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

BAB II TINJAUAN PUSTAKA

Bab ini membahas dasar pembuatan dan beberapa teori penunjang yang berhubungan dengan pokok pembahasan yang mendasari pembuatan tugas akhir ini.

BAB III PERANCANGAN SISTEM

Bab ini membahas perancangan dari sistem yang akan diterapkan. Terdiri dari alur kerja program, penggunaan dataset, dan desain arsitektur CNN.

BAB IV IMPLEMENTASI

Bab ini membahas implementasi dari rancangan sistem yang dilakukan pada tahap perancangan. Penjelasan implementasi meliputi implementasi praproses, implementasi augmentasi dataset, dan implementasi arsitektur CNN.

BAB V ANALISA DAN UJI COBA

Bab ini membahas tahap-tahap pengujian dari sistem yang telah dirancang. Bab ini meliputi persiapan pengujian dan hasil uji coba beserta analisanya.

BAB VI KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dari hasil uji coba yang telah dilaksanakan beserta saran untuk penelitian selanjutnya.

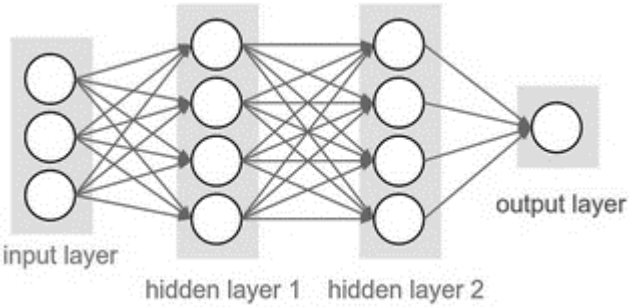
BAB II
TINJAUAN PUSTAKA

Bab ini berisi penjelasan teori dan materi yang berkaitan dengan algoritma yang diajukan. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap program yang dibuat dan berguna sebagai penunjang dalam pengembangan perangkat lunak.

2.1 Convolutional Neural Network (CNN)

CNN seperti metode *neural network* pada umumnya, terbentuk oleh berlapis-lapis neuron yang memiliki berat (*weight*) dan bias yang dapat diatur. Setiap neuron memperoleh masukan dari layer *input*, melakukan *dot-product* pada layer-layer berikutnya dan menghasilkan keluaran pada layer *output*. Pada umumnya semakin banyak jumlah layer, semakin tinggi akurasi dan semakin kompleks kemampuan dari jaringan, namun pada suatu titik akan terjadi *diminishing return*, dimana peningkatan jumlah layer tidak meningkatkan kinerja dari jaringan [4]. Network ini dilatih untuk mengambil data mentah dan membuat korelasi antara data tersebut terhadap skor yang diperoleh di akhir. Semakin tinggi skor akhir, semakin bias jaringan terhadap konfigurasi tertentu yang menghasilkan skor tersebut. Arsitektur *neural network* diilustrasikan pada Gambar 2.1.

CNN merupakan *deep neural network*, arsitektur neural network yang terdiri dari 2 atau lebih dari 2 hidden layer. Ciri khas dari CNN adalah dalam CNN data masukan selalu berukuran 2 dimensi dan berupa gambar, sehingga ada properti-properti tertentu yang dapat di-*encode* ke dalam arsitektur *network* [5].

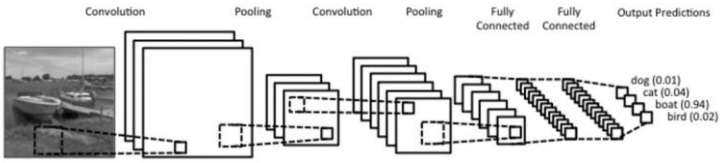


Gambar 2.1 Ilustrasi arsitektur neural network [5]

Gambar yang dimasukkan ke dalam CNN sebaiknya di-*downscale* terlebih dahulu. Gambar yang berukuran besar akan menghasilkan jumlah piksel sebanyak hasil perkalian dari panjang piksel, lebar piksel, dan *color channe*l dari gambar. Dengan melakukan *downscaling* akan mempersingkat waktu yang diperlukan untuk melatih jaringan nantinya [5].

CNN secara umum terdiri dari 3 jenis layer neuron [5], yaitu *Convolutional Layer, Pooling Layer, and Fully-Connected Layer*. Layer ini kemudian ditumpuk untuk membentuk sebuah arsitektur CNN yang lengkap. Pada layer pertama *Convolution Layer*, memiliki filter yang berdimensi kecil yang kemudian dikonvolusikan di atas gambar (direpresentasikan ke dalam bentuk matriks) yang masuk. Ketika berada di atas piksel tertentu, filter kemudian menghasilkan keluaran berupa matriks citra. Layer kedua, *Pooling Layer* melakukan *downsampling* terhadap data untuk mengurangi harga komputasi dalam memproses matriks. Pada layer ketiga *Fully-Connected Layer*, merupakan layer yang memiliki koneksi penuh terhadap layer-layer sebelumnya, layer

ini sama seperti pada *neural network* biasanya. Ilustrasi alur kerja layer CNN dapat dilihat pada Gambar 2.2.



Gambar 2.2 Ilustrasi alur kerja CNN [6]

2.1.1 Layer pada CNN

Pada CNN, terdapat 3 jenis layer yang digunakan, yaitu:

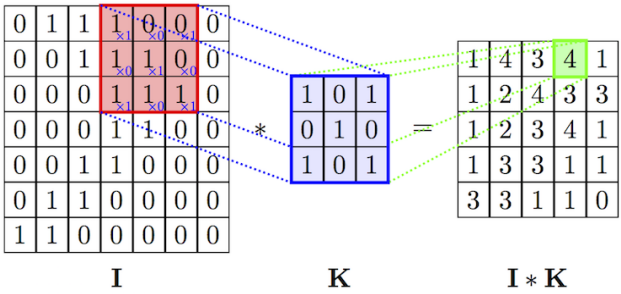
1. *Convolution Layer*

   Merupakan tulang belakang dari CNN, layer konvolusi mengambil data citra, dan mengaplikasikan sebuah kernel/filter yang bergeser diatas citra. Kernel tersebut kemudian melakukan ekstraksi fitur dengan cara melakukan dot-product. Hasil dari dot-product ini kemudian dapat diserahkan ke layer selanjutnya. Setiap kernel dari layer konvolusi memiliki weight sebesar ukuran dari kernel tersebut yang kemudian akan diubah-ubah menggunakan *backpropagation*.

   Satu layer konvolusi dapat memiliki lebih dari satu jenis kernel, dimana masing-masing kernel memiliki weight yang berbeda dan mengekstraksi fitur yang berbeda pula. Seluruh kernel ini kemudian akan membentuk citra baru (citra hasil perkalian dot-product) dengan kedalaman channel sesuai dengan jumlah kernel [5].

   Layer konvolusi menerima input dengan ukuran 2-d spasial,(matrix) sedangkan layer *fully-*

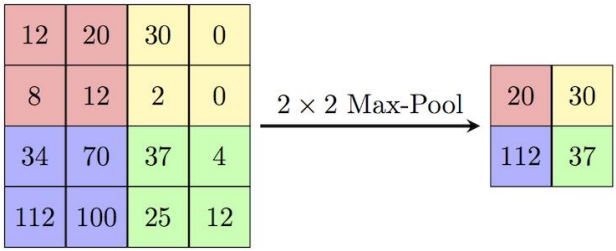
connected hanya menerima input dengan ukuran 1-d spasial (array). Gambar 2.3 mengilustrasikan cara kerja filter pada layer konvolusi.



Gambar 2.3 Layer konvolusi [7]

2. *Pooling Layer*

Layer Pooling digunakan untuk *downsampling* matrix gambar, sehingga harga dari arsitektur berkurang. Layer ini tidak memiliki weight, dan hanya berlaku sebagai filter. Gambar 2.4 mengilustrasikan cara kerja filter pada layer maximum pooling.



Gambar 2.4 Layer pooling [7]

3. *Fully-Connected Layer* (FC Layer)

FC Layer pada CNN sama pada dasarnya dengan FC yang terdapat pada neural network lainnya. FC dan Convolution layer memiliki fungsi yang mirip dalam melakukan *training* dan menyimpan weight dari model. Namun, dikarenakan hanya mendukung data

dengan 1 dimensi spasial, FC sebaiknya hanya diterapkan diakhir arsitektur. Ini guna mencegah kehilangan data akibat mengubah data gambar 2-d spasial menjadi data 1-d spasial.

4. *Padding Layer*

Ketika konvolusi dilakukan, ukuran citra akan mengecil. Untuk mempertahankan ukuran citra, maka dilakukan padding pada citra menggunakan *Padding Layer*. Paddding layer tidak menyimpan weight dari neural network. Hasil keluaran layer *padding* dapat dilihat pada Gambar 2.5

0	0	0	0	0	0	0	0
0	18	54	51	239	244	188	0
0	55	121	75	78	95	88	0
0	35	24	204	113	109	221	0
0	3	154	104	235	25	130	0
0	15	253	225	159	78	233	0
0	68	85	180	214	245	0	0
0	0	0	0	0	0	0	0

Gambar 2.5 Layer padding [8]

5. *Dropout Layer*

Dropout layer dimasukkan untuk mematikan neuron secara random pada layer sebelumnya (biasanya ditaruh setelah fully-connected layer). Penggunaan dropout layer umumnya dilakukan untuk mencegah overfitting akibat neuron yang terlalu bergantung antar satu sama lain. Akibatnya, dihasilkan model yang lebih independent dan fleksibel, dengan performa *training* yang meningkat [9].

2.1.2 Fungsi Aktivasi

Fungsi aktivasi adalah sebuah fungsi yang mengambil nilai input dari sebuah neuron dan mengeluarkan nilai output yang kemudian dilanjutkan ke neuron lain. Fungsi ini menambah properti non-linear kedalam algoritma, Properti non-linear ini penting agar algoritma dapat menyelesaikan permasalahan kompleks [10]. Tanpa properti ini, algoritma tidak akan dapat menyelesaikan permasalahan yang kompleks, menghasilkan akurasi yang buruk, dan menghasilkan algoritma CNN yang tidak fleksibel karena bersifat layaknya seperti *single-layer perceptron* [7] [11]. Beberapa fungsi aktivasi yang sering digunakan dalam CNN adalah sebagai berikut:

a. *Sigmoid*

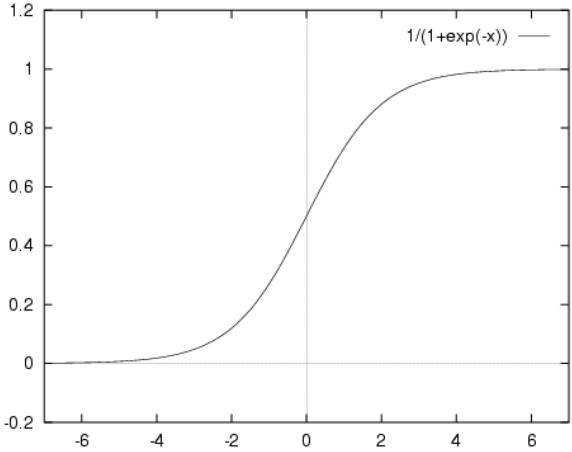
Fungsi aktivasi sigmoid memiliki bentuk seperti pada persamaan 2.1.

$$\sigma(x) = \frac{1}{(1+e^{-x})} \tag{2.1}$$

Memiliki rentang antara 0 dan 1 dan memiliki bentuk kurva S yang dapat dilihat pada Gambar 2.6. Fungsi ini mudah dimengerti dan diterapkan, namun sudah jarang digunakan, dikarenakan beberapa alasan sebagai berikut:

- Gradien data hilang (*Vanishing Gradient Problem*)
- Titik tengah kurva dari fungsi ini bukan 0. Sehingga output tidak konsisten dan bisa bergeser terlalu jauh ke atas atau kebawah. Ini mengakibatkan fungsi sigmoid sulit dioptimasi.
- Gradien tersaturasi, merupakan salah satu permasalahan terbesar, dimana lama-kelamaan data memiliki tendensi untuk berkumpul mendekati titik 0 atau 1, sehingga keragaman output weight neuron kecil.

- Konvergensi yang lambat



Gambar 2.6 Grafik kurva Sigmoid [12]

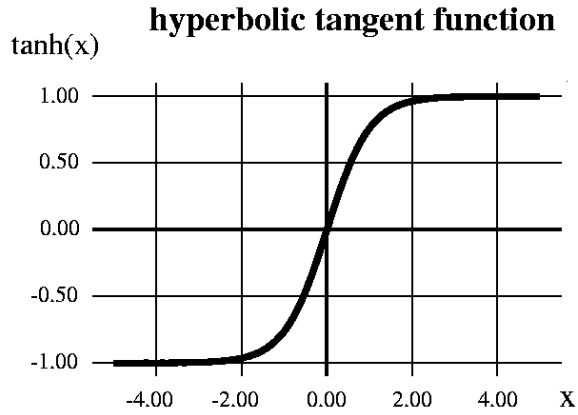
b. Tanh (Hyperbolic Tangent)

Merupakan fungsi aktivasi untuk menangani permasalahan dari sigmoid. Memiliki bentuk seperti pada persamaan 2.2.

$$\sigma(x) = \tanh(x) \tag{2.2}$$

Fungsi Tanh merupakan peningkatan dari fungsi sigmoid. Fungsi Tanh memiliki titik tengah 0, sehingga menyelesaikan salah satu permasalahan fungsi sigmoid yaitu di permasalahan optimasi fungsi aktivasi.

Fungsi aktivasi Tanh lebih mudah dioptimasi, namun permasalahan gradien tersaturasi dan *vanishing gradient* masih tetap ada. Oleh karena itu fungsi Tanh jarang digunakan. Grafik kurva Tanh dapat dilihat pada Gambar 2.7.



Gambar 2.7 Grafik kurva Tanh [12]

c. *ReLU* (*Rectified Linear Unit*)

Fungsi aktivasi yang sangat populer beberapa tahun terakhir, memiliki bentuk seperti pada persamaan 2.3.

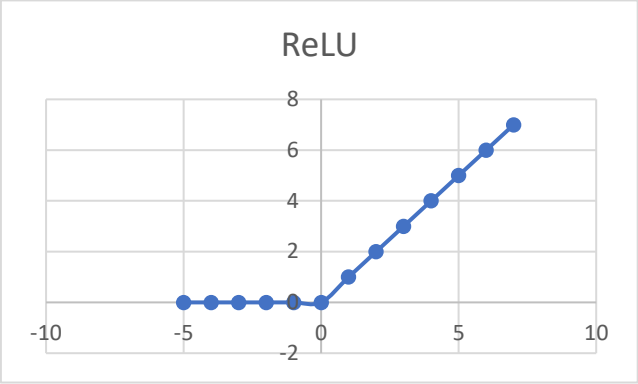
$$\sigma(x) = \max(0, x) \tag{2.3}$$

Dikarenakan fungsinya yang sangat sederhana, fungsi ReLU memiliki performa yang lebih tinggi dibandingkan fungsi aktivasi lainnya. Selain itu juga ada beberapa hal lain yang menyebabkan fungsi ReLU lebih dipilih dalam membangun neural network:

- Konvergensi data yang cepat
- Pergerakan data stabil, tidak mengalami permasalahan gradien hilang
- *Cost-effective*
- Mudah dioptimasi.

Namun fungsi ReLU memiliki kekurangan yaitu ketika banyak input yang negatif, semua output akan diberi nilai 0, sehingga neuron tidak

mengeluarkan informasi dan mati. Grafik kurva ReLU dapat dilihat pada Gambar 2.8.



Gambar 2.8 Grafik kurva ReLU

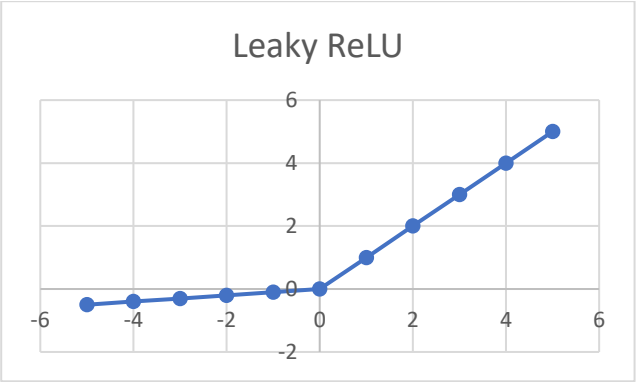
d. Leaky ReLU

Untuk menyelesaikan permasalahan neuron mati dari ReLU, maka digunakan Leaky ReLU dengan bentuk seperti pada persamaan 2.4 berikut ini.

$$\sigma(x) = \max(x/100, x) \tag{2.4}$$

Fungsi Leaky ReLU memungkinkan nilai negatif untuk dikeluarkan oleh ReLU. Nilai negatif yang dikeluarkan dibagi dengan konstanta tertentu sehingga nilainya menjadi kecil.

Sifat dari Leaky ReLU adalah konvergensi positif yang cepat, namun konvergensi negatif yang lambat. Ini menyebabkan neuron dengan input negatif masih dapat mengeluarkan output dan tidak mati [7]. Gambar 2.9 menggambarkan grafik kurva dari fungsi aktivasi LeakyReLU



Gambar 2.9 Grafik kurva Leaky ReLU

e. Softmax

Fungsi aktivasi softmax umumnya digunakan pada layer terakhir dari sebuah neural network untuk menghitung dan mencari probabilitas kelas target. Memiliki rentang 0 hingga 1. Fungsi softmax menghitung nilai probabilitas kelas target dan nilai probabilitas dari kelas lain, dengan jumlah nilai probabilitas 1 untuk seluruh persebaran kelas. Semakin besar nilai softmax, semakin tinggi probabilitas suatu data merupakan bagian dari kelas tersebut. Fungsi softmax dapat dihitung menggunakan persamaan 2.5.

$$\sigma(Xi) = \frac{e^{Xi}}{\sum_{j=0}^{k} e^{Xj}} \quad (2.5)$$

2.2 Facial Landmark Estimation and Alignment

Citra wajah yang masuk kadang dalam posisi yang tidak ideal, sehingga dapat menurunkan performa algoritma yang digunakan dalam melakukan klasifikasi. Untuk mencegah hal tersebut, dilakukan *facial landmark estimation*, untuk memperoleh titik-titik fitur pada citra wajah. Setelah

memperoleh titik ini dimulai proses *alignment* dengan cara menggeser titik-titik wajah pada citra ke posisi *canon* pada plot [13]. Pada tugas akhir ini, digunakan pendekatan yang diusulkan oleh Vahid Kazemi dan Josephine Sullivan [14], dengan implementasi menggunakan pustaka dlib [15] dan Openface [16].

2.3 Deep Face Recognition (VGG16)

Omkar M. Parkhi, Andrea Vedaldi, dan Andrew Zisserman, tergabung dalam Visual Geometry Group (VGG) dari Universitas Oxford telah mempublikasikan paper mengenai Face Recognition menggunakan CNN. Tujuan dari paper tersebut adalah menyediakan sebuah metode *deep learning face recognition* yang tidak memerlukan dataset yang banyak [17].

Arsitektur yang digunakan oleh Parkhi ini dalam perkembangannya kemudian disebut dengan nama VGGNet atau VGG. Terdapat 2 versi VGG yang umumnya dikenal yaitu VGG16 dan VGG19. Namun arsitektur yang secara terkhusus digunakan oleh paper referensi adalah arsitektur VGG16 yang terdiri dari:

- 5 blok layer konvolusi (1 blok terdiri dari setidaknya 2 layer konvolusi diikuti dengan sebuah layer pooling),
- 3 layer *fully-connected* (2 *dense layer* diikuti dengan 1 *classifier layer*)

VGG19 secara struktural memiliki arsitektur yang sama dengan VGG16 hanya saja terdapat penambahan jumlah layer konvolusi. Pada VGG19 ditambah tiga layer konvolusi setelah urutan blok konvolusi kedua.

2.4 Python

Bahasa pemrograman Python [18] memiliki beberapa keunggulan dibandingkan bahasa pemrograman lainnya:

- Memiliki *library* yang ekstensif, dalam distribusi Python telah disediakan modul-modul siap pakai untuk berbagai keperluan.
- Tidak ada deklarasi tipe sehingga program menjadi lebih sederhana, singkat dan fleksibel
- Adanya sistem indentasi untuk membuka dan menutup suatu fungsi sehingga code akan terlihat lebih rapi, mudah dibaca dan dikembangkan.
- Python menyediakan bahasa pemrograman optimasi untuk kegunaan bersama dengan perangkat bantu yang dibutuhkan untuk diintegrasikan dengan bahasa pemrograman lainnya.

2.5 Tensorflow

Tensorflow [19] adalah sebuah library open-source yang dikembangkan oleh Tim Google Brain. Awalnya dikembangkan untuk melakukan riset bersifat internal. Namun pada tahun 2015, dibuka untuk public dengan lisensi open-source. Tensorflow adalah sebuah tool untuk melakukan komputasi numerik dan machine learning (neural network, dan sebagainya). Tersedia untuk API Python dan C, Tensorflow dengan cepat digunakan oleh tim riset diseluruh dunia dikarenakan keunggulannya sebagai berikut:

- Dapat mengalokasikan resource untuk banyak CPU/GPU.
- Skalabilitas tinggi
- Penggunaan GPU transparan

- Optimasi pada komputasi matrix sehingga lebih efisien dalam penggunaan memori.

2.6 Keras

Keras adalah library neural network open-source yang ditulis menggunakan Bahasa pemrograman Python. Keras berdiri diatas Tensorflow (juga dapat menggunakan Theano, MXNet, atau Microsoft Cognitive Toolkit) sehingga merupakan Bahasa pemrograman tingkat tinggi, yang mempermudah user melakukan eksperimentasi neural network.

Keras mengandung berbagai macam implementasi tool pembangun neural network, misalnya untuk membangun layer, objektif, fungsi aktivasi, dan sebagainya. Selain itu Keras juga mensupport utilisasi GPU [20].

2.7 Akurasi

Nilai akurasi merupakan persentase kejadian benar (*True Positive* dan *True Negative*) dari seluruh prediksi yang dilakukan oleh model [21]. Perhitungan akurasi akan digunakan sebagai indikator ketepatan model CNN dalam mendeteksi wajah. Perhitungan akurasi dilakukan menggunakan persamaan 2.6.

$$\text{Akurasi} = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.6}$$

Metrik akurasi merupakan metrik standar yang dapat digunakan pada pustaka keras dalam membantu user mengevaluasi kinerja *neural network* [22].

2.8 CUDNN

Library Nvidia CUDA Deep Neural Network (cuDNN) [23] adalah library yang diakselerasi oleh GPU untuk implementasi neural network menggunakan Tensorflow, Theano, dan pustaka lainnya. Implementasi Deep Learning menggunakan GPU jauh lebih cepat dibandingkan implementasi menggunakan CPU, dikarenakan arsitektur GPU lebih baik dalam menangani jumlah neuron yang banyak [24].

2.9 *Hyperparameter Tuning*

Karena dalam *machine learning* terdapat banyak sekali parameter dalam pembuatan model, maka dikenal sebuah istilah yaitu hyperparameter. Hyperparameter didefinisikan sebagai parameter dari sebuah distribusi di luar distribusi pada model. Pada NN, yang disebut parameter adalah nilai bobot dan bias dari *layer*. Seluruh parameter lain disebut dengan istilah hyperparameter karena berada di luar domain distribusi dari model.

Hyperparameter tuning adalah istilah yang digunakan ketika pengguna mengubah satu atau lebih hyperparameter dari sistem, untuk memaksimalkan hal yang diinginkan dari network. Hal yang diinginkan ini bisa berupa akurasi, performa, maupun keduanya [25].

2.10 *Vanishing Gradient*

Vanishing gradient adalah istilah yang digunakan untuk mendeskripsikan fenomena yang sering terjadi pada deep neural network, dimana network tidak dapat belajar/lambat belajar. Ini dapat disebabkan oleh karena fungsi aktivasi yang digunakan memiliki tendensi untuk mengumpulkan data di

suatu titik. Ini dapat dilihat pada fungsi aktivasi *sigmoid* dan tanh.

Umumnya jika network tidak terlalu dalam, masalah ini dapat diselesaikan dengan menggunakan fungsi aktivasi yang memiliki sedikit linearitas seperti ReLU dibandingkan fungsi aktivasi *sigmoid*/tanh. Namun jika network terlalu dalam, kemungkinan ReLU dapat "membunuh" neuron, sehingga perlu digunakan Leaky ReLU untuk menggantikan ReLU. Fungsi aktivasi Leaky ReLU dapat membantu, dikarenakan fungsi aktivasi ini mencegah kondisi dimana neuron menjadi nonaktif dikarenakan terus mendapatkan input negatif.

Menurut paper yang dipublikasikan Bing Xu dan rekan [11], fungsi aktivasi Leaky ReLU berkali-kali mengungguli ReLU dalam masalah kecepatan *training* network. Walaupun begitu, keunggulan dari reproduksi penelitian ini masih belum disertai dengan suatu teori yang kuat, sehingga masih diperlukan investigasi lebih lanjut.

[Halaman ini sengaja dikosongkan]

BAB III
PERANCANGAN SISTEM

Pada bab ini akan dijelaskan mengenai desain dari perangkal lunak, beserta flowchart, diagram, dan data yang digunakan dalam implementasi algoritma.

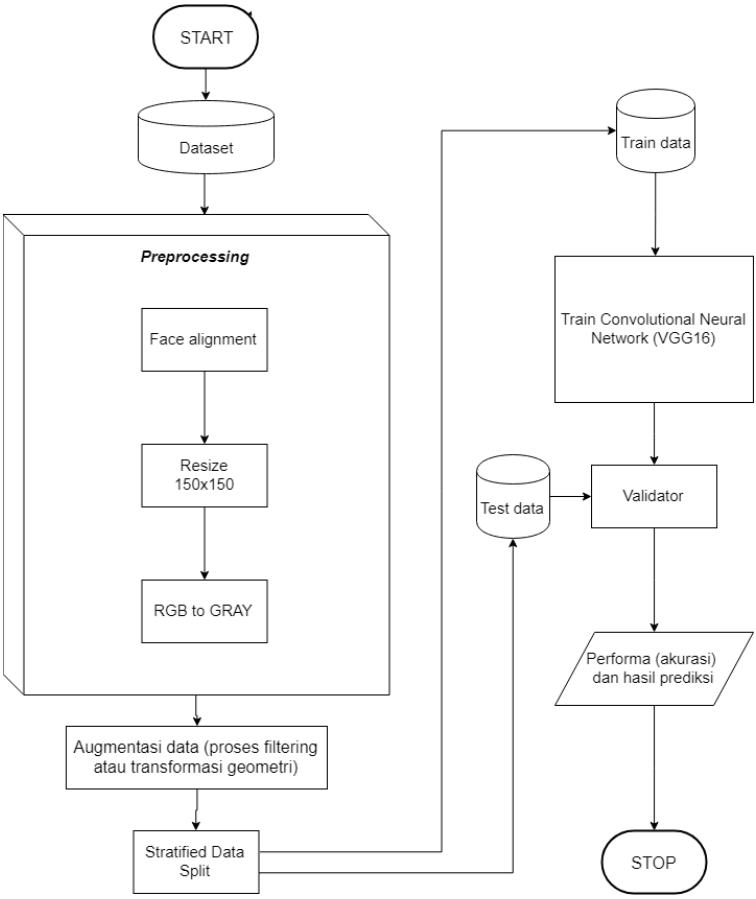
Pada tugas akhir ini, secara umum akan dilakukan persiapan dataset, perancangan arsitektur CNN, *feeding* dataset kedalam model, dan pengukuran performa model/neural network.

3.1 Alur Kerja Program

Alur kerja program mengikuti beberapa proses yang akan dijelaskan pada subbab ini. Pada awalnya, dataset akan dipraproses terlebih dahulu. Praproses ini terdiri dari beberapa langkah yaitu penerapan face alighment, pengecilan atau pengubahan ukuran gambar, dan konversi gambar dari yang awalnya RGB (3 channel warna) menjadi hitam putih (1 channel warna). Kemudian akan dilakukan augmentasi dataset menggunakan salah satu dari 2 metode augmentasi dataset (augmentasi data dengan proses *filtering* dan augmentasi data dengan proses transformasi geometri) yang akan diimplementasikan untuk mengetahui dampaknya terhadap neural network.

Setelah augmentasi selesai dilakukan, dataset akan dibagi menjadi *training data* dan *testing data* dengan rasio perbandingan 70% *training* dan 30% *testing*, menggunakan metode *Stratified Shuffle Split*. *Training data* atau *training set* akan digunakan untuk melatih CNN, sedangkan *testing data* atau *testing set* akan digunakan untuk menguji kemampuan CNN dalam mengenali wajah. Sistem akan mengeluarkan nilai akurasi sebagai hasil akhir dari program, disertai dengan hasil

prediksi gambar. Diagram alur kerja sistem dapat dilihat pada Gambar 3.1.



Gambar 3.1 Alur kerja sistem.

3.2 Dataset

Berikut ini adalah dataset yang akan digunakan di dalam tugas akhir.

1. Dataset faces94 [3].

   Berisi 3.059 citra wajah dari 153 orang (kelas). Dataset ini memilki persebaran data yang konsisten dimana setiap kelas terdiri dari kurang lebih 20 citra. Selain persebaran data yang konsisten, kelebihan dataset ini dibandingkan LFW adalah citra wajah selalu menghadap kedepan, dan selalu terdapat 1 wajah didalam 1 citra wajah.

   Dataset ini akan digunakan untuk menguji kecepatan *learning* dari sistem karena ukuran dataset yang lebih kecil dan citra wajah yang konsisten. Manipulasi dataset tidak diperlukan. Contoh dari dataset faces94 dapat dilihat pada Gambar 3.2.

Gambar 3.2 Dataset Faces94

2. Dataset LFW [2].

   Dataset ini terdiri dari 13.233 citra wajah dari 5.749 orang (kelas), namun memiliki persebaran data yang tidak merata. Citra wajah dalam dataset ini tidak selalu menghadap ke depan dan dalam satu citra tidak

selalu hanya terdiri dari 1 wajah (bisa saja terdiri dari 2 wajah), sehingga lebih sulit untuk dikenali oleh sistem dibandingkan dataset faces94.

Dataset ini akan digunakan sebagai penguji akurasi sistem, dikarenakan dataset ini lebih representatif terhadap kejadian didunia nyata. Dalam implementasinya dataset akan diambil dengan variasi sebagai berikut:

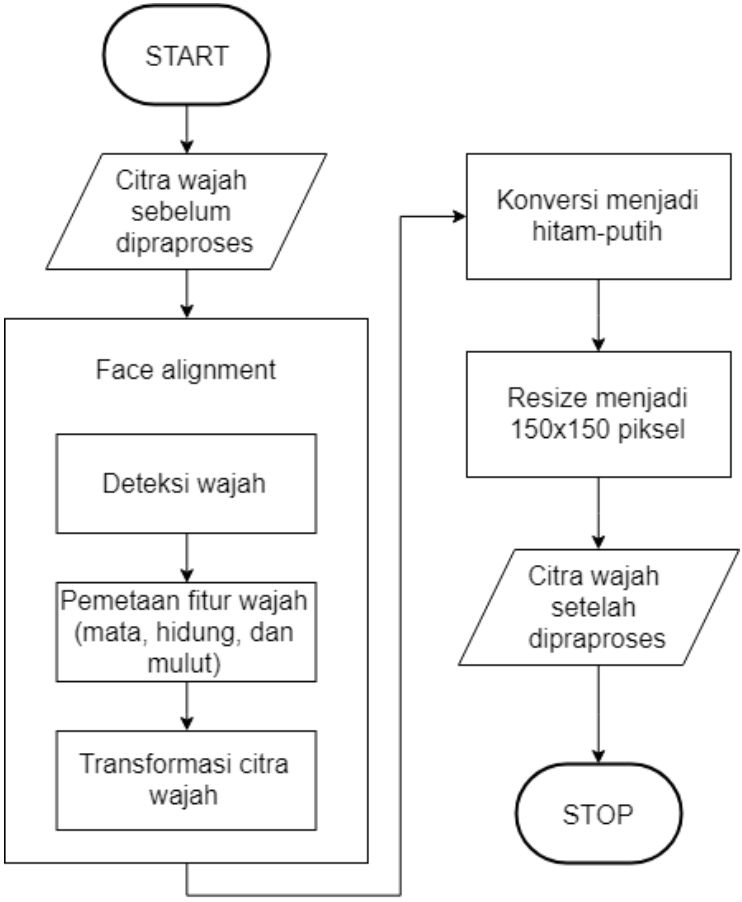
a. Variasi A terdiri dari 10 citra wajah per kelas, dengan cara menghapus kelas dengan citra wajah dibawah 10. Jika citra diatas 10, cukup ambil 10 citra pertama. Jumlah citra adalah 1.580 (158 kelas).
b. Variasi B terdiri dari 40 citra wajah per kelas, dengan cara menghapus kelas dengan citra wajah dibawah 40. Jika citra diatas 40, cukup ambil 40 citra pertama. Jumlah citra adalah 760 (19 kelas).
c. Variasi C terdiri lebih dari 40 citra wajah per kelas, dengan cara menghapus kelas dengan citra wajah dibawah 40. Jumlah citra adalah 1.867 (19 kelas).

Pengaruh dari variasi pengambilan data ini dapat dilihat pada Subbab 5.2.2. Contoh dataset LFW dapat dilihat pada Gambar 3.3.

Gambar 3.3 Dataset LFW

3.2.1 Praproses Dataset

Beberapa praproses yang dilakukan akan dijelaskan pada bagian ini, disertai dengan alur kerja yang dapat dilihat pada Gambar 3.4.



Gambar 3.4 Diagram alur kerja praproses

a. *Face Alignment*

Dataset yang masuk terlebih dahulu di praproses untuk menemukan wajah yang terdapat didalamnya. Untuk menemukan lokalisasi wajah, digunakan algoritma HOG (Histogram of Oriented Gradient) melalui pustaka Dlib. Setelah menemukan posisi wajah, dilakukan cropping pada gambar, untuk menghapus citra lain selain citra wajah yang ingin diproses. Jika terdapat lebih dari 1 wajah dalam satu gambar, maka ambil wajah terbesar. Ilustrasi proses ini dapat dilihat pada Gambar 3.5.

Gambar 3.5 Deteksi wajah menggunakan HOG

Pada langkah selanjutnya dilakukan *facial landmarking* atau pemetaan wajah, untuk menemukan fitur wajah (mata, hidung, mulut), menggunakan metode *Regression Tree* yang dikembangkan oleh Vahid Kazemi dan Josephine Sullivan [14]. Pustaka Openface, pustaka Python yang dikembangkan dan digunakan secara khusus untuk mengolah dan mengenali wajah akan digunakan untuk melakukan proses landmarking. Gambar 3.6 menunjukkan hasil deteksi fitur wajah (*landmarking*).

Gambar 3.6 Gambar wajah dan fitur yang dideteksi

Setelah landmarking dilakukan, kemudian gambar akan ditransformasi/*alignment* menggunakan translasi, rotasi, skala, dan *affine transformation*. Tidak digunakan transformasi 3-D agar tidak terjadi distorsi pada gambar. Hasil dari transformasi adalah posisi wajah akan menengah. Proses ini juga dilakukan menggunakan pustaka Openface. Gambar 3.7 menunjukkan hasil transformasi.

Gambar 3.7 Gambar wajah setelah ditransformasi

b. Menjadikan hitam-putih

Citra akan dikonversi menjadi hitam putih untuk mengurangi harga komputasi, seperti yang terlihat pada Gambar 3.8. Pustaka OpenCV digunakan dalam proses ini.

Gambar 3.8 Citra wajah hitam-putih

c. Resize

Citra akan diresize menjadi ukuran 150 x 150 piksel menggunakan pustaka OpenCV untuk mempersingkat waktu *training*, setelah dilakukan pengujian dalam tugas akhir ini, ukuran citra 224x224 (standar VGG) maupun 150x150 tidak memiliki perbedaan signifikan dalam menghasilkan model yang akurat. Namun penelitian lebih lanjut masih diperlukan.

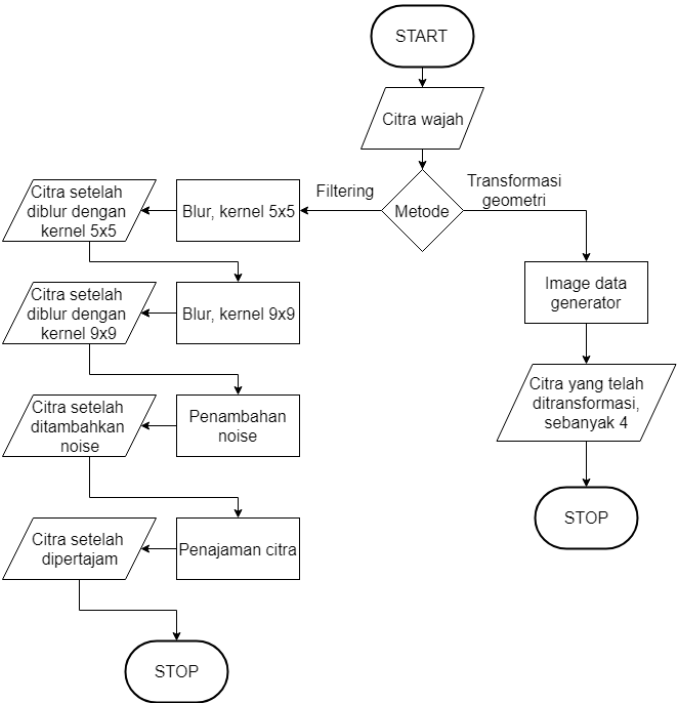
3.2.2 Augmentasi Dataset

Untuk pengenalan wajah, pada umumnya memerlukan jumlah data yang sangat banyak. Dikarenakan dataset yang tersedia secara bebas dan gratis di internet tidak terlalu banyak, maka perlu dilakukan augmentasi dataset untuk menghasilkan akurasi model yang tinggi. Diagram alur dari proses augmentasi dapat dilihat pada Gambar 3.9.

Dua metode augmentasi yang digunakan dalam tugas akhir adalah sebagai berikut:

1. Augmentasi data dengan proses *filtering*

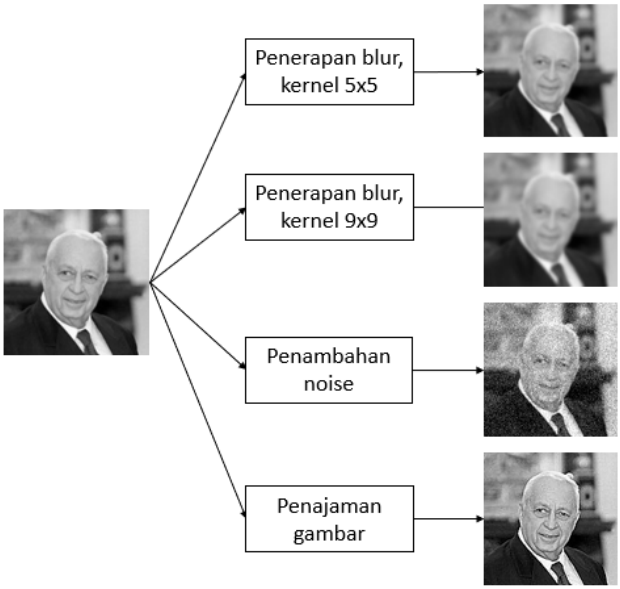
   Untuk setiap citra wajah, akan diterapkan 4 jenis filter yaitu *blur* dengan ukuran kernel 5x5, *blur* dengan kernel 9x9, penambahan *noise*, dan penajaman citra.



Gambar 3.9 Diagram alur augmentasi data citra

Pada filter blur, digunakan metode blur Gaussian menggunakan pustaka OpenCV. Sedangkan pada filter noise, dilakukan penambahan noise dengan cara menerapkan penjumlahan dan pengurangan secara random pada setiap piksel, dengan parameter standar deviasi 25. Penajaman citra dilakukan dengan menggunakan kernel penajaman dengan ukuran 3x3 dan parameter nilai intensitas 9, menggunakan pustaka OpenCV.

Setelah augmentasi ini, jumlah citra akan bertambah sebanyak 5 kali lipat. Ilustrasi hasil penerapan augmentasi filter dapat dilihat pada Gambar 3.10.



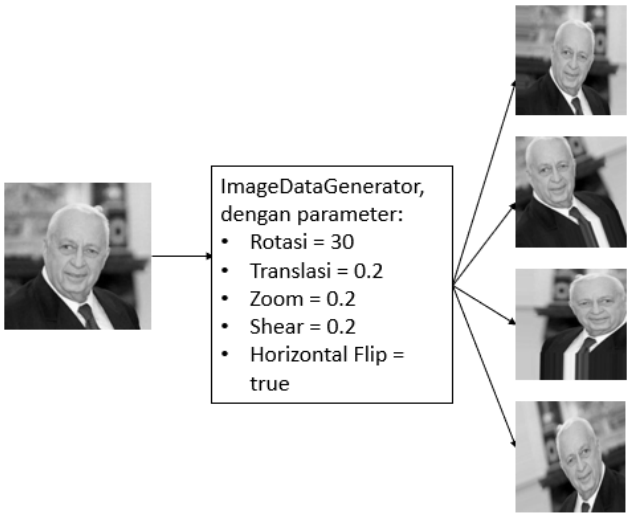
Gambar 3.10 Ilustrasi augmentasi filter

2. Augmentasi data dengan proses transformasi geometri

   Untuk setiap citra, akan diterapkan transformasi geometri berupa rotasi gambar, penggeseran gambar, pembesaran gambar, *shearing* gambar, dan *horizontal flip* secara random pada setiap gambar dengan rentang parameter yang telah ditetapkan.

   Rentang parameter sudut untuk transformasi rotasi adalah 30 derajat. Penggeseran gambar dilakukan dengan rentang penggeseran 20% dari ukuran awal gambar. Pembesaran gambar dilakukan dengan rentang pembesaran maksimal 0.2. Dan *shearing* gambar dilakukan dengan rentang parameter 0.2. Seluruh penerapan augmentasi diatas dicapai menggunakan pustaka ImageDataGenerator dari Keras.

Setelah augmentasi ini, jumlah citra akan bertambah sebanyak 5 kali lipat. Ilustrasi hasil penerapan augmentasi filter dapat dilihat pada Gambar 3.11.



Gambar 3.11 Ilustrasi augmentasi transformasi geometri

3.2.3 Pembagian *Training* dan *Testing* set

Ketika seluruh gambar dalam dataset telah dipraproses, gambar kemudian akan dibagi menjadi *training data*, dan *test data*. Metode pembagian yang digunakan adalah *Stratified Shuffle Split (Stratified Random Sampling)* [26] yang diimplementasikan menggunakan pustaka Scikit-Learn [27]. Metode *Stratified Shuffle Split* mengambil seluruh data, melakukan shuffle pada data, dan kemudian membagi data menjadi *training* dan *testing* set untuk setiap kelas. Rasio pembagian *training* dan *testing* set adalah 70:30.

Rangkaian proses pembagian data ini untuk memastikan representasi *training data* yang akurat terhadap *testing data*, oleh sebab itu tidak perlu dilakukan *tuning* pada nilai *seed*. *Training data* atau *training set* adalah data-data yang akan digunakan untuk melatih model. Network akan secara aktif mengubah *weight*-nya guna memaksimalkan akurasi dan meminimalkan *loss* dari *training data*. Sedangkan, *testing data* atau *testing set* adalah data-data yang digunakan untuk menguji akurasi dari model yang telah dibuat.

3.3 Desain Arsitektur CNN

Desain arsitektur neural network yang digunakan pada tugas akhir pengenalan wajah ini akan dibahas dibagian ini. Arsitektur yang digunakan pada paper referensi adalah arsitektur VGG16. Namun pada tugas akhir dilakukan modifikasi pada arsitektur, yaitu mengganti metode aktivasi dari ReLU menjadi LeakyReLU. Ini terbukti meningkatkan kecepatan *training* dari neural network pada tugas akhir (dapat dilihat pada hasil uji coba).

Pada pengujian, diatur jumlah batch sebanyak 6 karena keterbatasan memori GPU. Jumlah epoch/generasi mula-mula adalah 300, namun untuk menghemat waktu, pelatihan akan dihentikan sesuai kebutuhan ketika sudah terjadi konvergensi, walaupun jumlah epoch belum tercapai. Jika setelah 300 epoch belum terjadi konvergensi, pelatihan akan dihentikan dan akan diambil hasil pelatihan pada epoch ke 300. Pelatihan disebut konvergen jika sudah tidak terjadi perubahan pada nilai akurasi *training* (*plateau*) atau jika nilai akurasi *training* sudah mencapai 100%. Optimizer yang akan digunakan adalah optimizer SGD dengan parameter learning rate 0.001, momentum 0, dan nilai *decay* 0. Parameter SGD ini tidak diubah selama pengujian dilakukan.

Parameter arsitektur dan spesifikasi layer dapat dilihat pada Tabel 3.1 dan Tabel 3.2.

Tabel 3.1 Parameter Arsitektur

Parameter	Nilai
Jumlah Layer	16
Ukuran batch	6
Jumlah epoch	300
Optimizer	SGD (learning rate 0.001, momentum=0.0, decay=0.0)
Fungsi Aktivasi	LeakyReLU

Tabel 3.2 Layer Arsitektur VGG16

#Blok	Layer	Output Shape	Parameter #
1	Conv Layer 1, LeakyReLU	(150, 150, 64)	640
	Conv Layer 2, LeakyReLU	(150, 150, 64)	36,928
	Max Pooling Layer (Downscale 50%)	-	-
2	Conv Layer 3, LeakyReLU	(75, 75, 128)	73,856
	Conv Layer 4, LeakyReLU	(75, 75, 128)	147,584
	Max Pooling Layer (Downscale 50%)	-	-
3	Conv Layer 5, LeakyReLU	(37, 37, 256)	295,168
	Conv Layer 6, LeakyReLU	(37, 37, 256)	590,080
	Conv Layer 7, LeakyReLU	(37, 37, 256)	590,080
	Max Pooling Layer (Downscale 50%)	-	-
4	Conv Layer 8, LeakyReLU	(18, 18, 512)	1,180,160
	Conv Layer 9, LeakyReLU	(18, 18, 512)	2,359,808
	Conv Layer 10, LeakyReLU	(18, 18, 512)	2,359,808
	Max Pooling Layer (Downscale 50%)	-	-

5	Conv Layer 11, LeakyReLU	(9, 9, 512)	2,359,808
	Conv Layer 12, LeakyReLU	(9, 9, 512)	2,359,808
	Conv Layer 13, LeakyReLU	(9, 9, 512)	2,359,808
	Max Pooling Layer (Downscale 50%)	-	-
	Flatten Layer (2D to 1D)	(8192)	-
6	FC Layer 1, LeakyReLU	(4096)	33,558,528
7	FC Layer 2, LeakyReLU	(4096)	16,781,312
8	FC Layer 3, Softmax	(152) (num_classes)	622,744
	TOTAL (WEIGHT + BIAS)		65,676,120

BAB IV
IMPLEMENTASI

4.1 Implementasi Praproses

Untuk mengimplementasikan praproses, digunakan library OpenCV, Dlib, dan Openface. Template/model untuk face alignment disediakan oleh Dlib dan dapat didownload melalui halaman web berikut ini: http://dlib.net/files/shape_predictor_68_face_landmarks.dat.bz2

Implementasi praproses diawali dengan mengambil template *face alignment* dari halaman web di atas, dan memuatnya menggunakan pustaka Openface. Setelah itu dilakukan deteksi wajah dengan metode HOG, menggunakan pustaka Dlib. Setelah wajah didapatkan, akan dilakukan pencarian fitur wajah menggunakan pustaka Openface dan pustaka Dlib. Fitur-fitur wajah yang sudah didapatkan kemudian akan digeser atau ditransformasi menggunakan template yang sudah dimuat diawal. Penggeseran yang dilakukan antara lain adalah translasi, skala, rotasi, dan affine transformation. Semua transformasi ini dilakukan sesuai dengan kebutuhan untuk menghasilkan citra wajah yang menengah. Setelah dilakukan *face alignment*, dihasilkan dataset citra wajah yang tidak terganggu oleh *background* dan menengah. Kemudian gambar akan diresize menjadi ukuran 150x150 menggunakan pustaka OpenCV. Dan pada akhirnya gambar akan dikonversi menjadi gambar hitam-putih, menggunakan pustaka OpenCV. Potongan kode 4.1 berikut adalah potongan kode untuk melakukan implementasi praproses data menggunakan bahasa pemrograman Python.

```
predictor_model                                                    =
"model/shape_predictor_68_face_landmarks.dat"

# create a HOG face detector using the built-in dlib class
face_detector = dlib.get_frontal_face_detector()
face_aligner = openface.AlignDlib(predictor_model)

for file in imlisting:
    im = cv2.imread(path1+'\\'+file)

    # run the HOG face detector on the image data
    detected_faces = face_detector(im, 1)

    print("Found    {}    faces    in    the    image    file
{}".format(len(detected_faces), file))

    # loop each faces in the photo
    face_rect = list(detected_faces)
    if len(face_rect)<1:
        # if no face is found, resize the image
        print("No face is found, resizing image")
        alignedFace = cv2.resize(im, (150, 150))
    elif len(face_rect)>1:
        # if more than 1 face, find the biggest face
        area = [this.area() for this in face_rect]
        max_area = max(area)
        max_index = area.index(max_area)
        alignedFace     =     face_aligner.align(150,     im,
face_rect[max_index],
landmarkIndices=openface.AlignDlib.OUTER_EYES_AND_NOSE)
    else:
        alignedFace     =     face_aligner.align(150,     im,
face_rect[0],
landmarkIndices=openface.AlignDlib.OUTER_EYES_AND_NOSE)

    # convert to BW
    alignedFace          =          cv2.cvtColor(alignedFace,
cv2.COLOR_BGR2GRAY)

    # save preprocessed image
    cv2.imwrite(path2+'\\'+file, alignedFace)
```

Potongan Kode 4.1 Implementasi praproses

4.2 Implementasi Augmentasi Dataset

Augmentasi yang dilakukan pada gambar terdiri dari 2 jenis augmentasi, yaitu augmentasi menggunakan proses filtering, dan augmentasi menggunakan proses transformasi geometri. Untuk augmentasi dengan proses filtering, digunakan pustaka OpenCV dalam implementasinya, sedangkan untuk augmentasi dengan proses transformasi geometri, digunakan pustaka ImageDataGenerator dari Keras dalam implementasinya.

Seluruh parameter yang digunakan dalam implementasi fungsi dijelaskan pada SubBab 3.2.2 mengenai Augmentasi Dataset. Potongan Kode 4.2 dibawah merupakan potongan kode yang memuat pseudocode dari proses augmentasi data dengan filtering. Sedangkan Potongan Kode 4.3 memuat pseudocode dari proses augmentasi data menggunakan transformasi geometri.

```
path = “dataset”
imagelist = listdir(path)

For im in imagelist
        image = loadImage(im)
        image_blur1 = image.addblur(kernel_size=5)
        image_blur2 = image.addblur(kernel_size=9)
        image_noise = image.addnoise(std=25)
        image_sharpen = image.sharpen(intensity=9)
        write(image_blur1, path)
        write(image_blur2, path)
        write(image_noise, path)
        write(image_sharpen, path)
```

Potongan Kode 4.2 Implementasi augmentasi data dengan proses filtering

```
path = “dataset”
imagelist = listdir(path)

Transformation = Datagen(zoom = 0.2, rotation = 30,
translation = 0.2, shear = 0.2, flip = true)

For im in imagelist
        Image = loadImage(im)
        For numberofcopy < 5
                Transformation.apply(random)
                Transformation.save(path)
```

Potongan Kode 4.3 Implementasi augmentasi data dengan proses transformasi geometri

4.3 Implementasi Arsitektur CNN (VGG16)

Menggunakan pustaka Keras, berikut adalah langkah implementasi arsitektur CNN (VGG16) beserta penjelasannya, dengan layer aktivasi Leaky ReLU. Leaky ReLU digunakan karena memiliki performa lebih tinggi seperti yang akan ditunjukkan pada Bab V Hasil Uji Coba.

Pertama, definisikan model menggunakan mode sekuensial. Mode sekuensial pada Keras memungkinkan pembuatan layer yang lebih fleksibel dan intuitif.

```
model = Sequential()
```

Pada VGG16, layer pertama yang ditambahkan adalah layer konvolusi dengan kedalaman neuron 64, ukuran kernel 3x3, padding 1 (same), dan stride 1. Tambahkan layer menggunakan kode berikut:

```
model.add(Conv2D(64, (3, 3), padding='same',
input_shape=(m,n,1)))
```

Parameter stride tidak dimasukkan (dibiarkan kosong), karena Keras menginisialisasi parameter stride dengan nilai 1 secara default. Sedangkan parameter padding di set menjadi ‘same’. Input shape adalah resolusi gambar yang akan dimasukkan beserta jumlah channelnya.

Setelah layer konvolusi, masukkan fungsi aktivasi nonlinear LeakyReLU pada network.

```
model.add(LeakyReLU())
```

Setelah fungsi aktivasi dimasukkan, tambahkan layer konvolusi selanjutnya, dengan kedalaman neuron 64, kernel 3x3, padding 1 (same), dan stride 1 (dibiarkan kosong), beserta fungsi aktivasi LeakyReLU.

```
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(LeakyReLU())
```

Setelah melakukan 2 kali konvolusi, lakukan downsampling menggunakan layer MaxPooling. Ukuran pooling adalah 2x2.

```
model.add(MaxPooling2D(pool_size=(2, 2)))
```

Setelah layer pooling ditambahkan, Blok konvolusi pertama telah selesai. Selanjutnya tambahkan layer konvolusi dan pooling untuk blok 2, blok 3, blok 4, dan blok 5, seperti contoh berikut.

```
#block1
model.add(Conv2D(64,        (3,        3),        padding='same',
input_shape=(m,n,1)))
model.add(LeakyReLU())
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(MaxPooling2D(pool_size=(2, 2)))
#block2
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(LeakyReLU())
```

```
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(MaxPooling2D(pool_size=(2, 2)))
#block3 dan seterusnya
#...
```

Selanjutnya akan ditambahkan 2 blok *fully-connected* (*Dense*) dengan parameter kedalaman neuron 4096. Namun sebelum itu, gambar perlu di *reshape* dari 2-D menjadi 1-D menggunakan layer *Flatten*. Berikut adalah contohnya.

```
model.add(Flatten())
#block6
model.add(Dense(4096))
model.add(LeakyReLU())
#block7
model.add(Dense(4096))
model.add(LeakyReLU())
```

Dan pada lapisan terakhir, ditambahkan layer *fully-connected* dengan kedalaman neuron sesuai dengan jumlah kelas dari dataset, dengan fungsi aktivasi softmax.

```
model.add(Dense(num_classes, activation='softmax'))
```

Potongan kode dibawah dijalankan ketika neural network sudah selesai dibangun:

```
opt = SGD(lr=1e-3)
model.compile(loss='categorical_crossentropy',   optimizer=opt,
metrics=['accuracy'])
```

Parameter loss digunakan *categorical_crossentropy* dikarenakan dataset yang digunakan terdiri lebih dari 2 kelas. Optimizer yang digunakan adalah SGD dengan learning rate 0,001. Metrics adalah variabel yang digunakan untuk menilai performa dari network.

Potongan Kode 4.4 memuat potongan kode lengkap dari arsitektur VGG16 yang dibangun.

```
model = Sequential()
model.add(Conv2D(64,      (3,      3),      padding='same',
input_shape=(m,n,1)))
model.add(LeakyReLU())
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(MaxPooling2D(pool_size=(2, 2))
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(256, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(Conv2D(256, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(Conv2D(256, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(512, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(Conv2D(512, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(Conv2D(512, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(512, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(Conv2D(512, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(Conv2D(512, (3, 3), padding='same'))
model.add(LeakyReLU())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(4096))
model.add(LeakyReLU())
model.add(Dense(4096))
model.add(LeakyReLU())
model.add(Dense(num_classes, activation='softmax'))
opt = SGD(lr=1e-3)
model.compile(loss='categorical_crossentropy',
optimizer=opt, metrics=['accuracy'])
```

Potongan Kode 4.4 Implementasi arsitektur VGG16

[Halaman ini sengaja dikosongkan]

BAB V
ANALISA DAN UJI COBA

5.1 Persiapan Uji Coba

Pada bagian berikut ini akan dipaparkan skenario pengujian yang akan dilakukan beserta lingkungan perangkat yang digunakan selama pengujian.

5.1.1 Skenario Uji Coba

Pada tugas akhir ini akan dilakukan pengujian berdasarkan tiga skenario besar uji coba berikut ini:

1. Skenario Pengujian 1:
   Uji coba perbandingan penggunaan fungsi aktivasi ReLU dan LeakyReLU pada VGG16, menggunakan dataset faces94.
2. Skenario Pengujian 2:
   Uji coba arsitektur VGG16 terbaik dari hasil no.1 untuk dataset LFW dengan jumlah citra bervariasi terhadap akurasi sistem.
3. Skenario Pengujian 3:
   Uji coba pengaruh augmentasi data (dengan proses *filtering* dan transformasi geometri) dan face alignment terhadap akurasi sistem, pada dataset LFW.

5.1.2 Lingkungan Uji Coba

Uji Coba dilakukan menggunakan bahasa pemrograman Python, dengan menggunakan pustaka Keras yang berdiri diatas Tensorflow. Untuk memastikan hasil uji coba yang didapatkan konsisten dan *reproducible*, didefinisikan seed yang sama pada setiap nilai random yang dihasilkan oleh sistem.

Spesifikasi perangkat, baik perangkat lunak (OS dan *software*), maupun perangkat keras (*hardware*) yang digunakan dapat dilihat pada Tabel 5.1

Tabel 5.1 Spesifikasi perangkat

Perangkat Lunak	
OS	Windows 10 - 64 Bit
IDE	Spyder
Perangkat Keras	
CPU	Intel Core i7 – 4710HQ
RAM	8 GB
GPU	Nvidia GTX 850M
GPU Memory	4 GB

5.2 Hasil Uji Coba

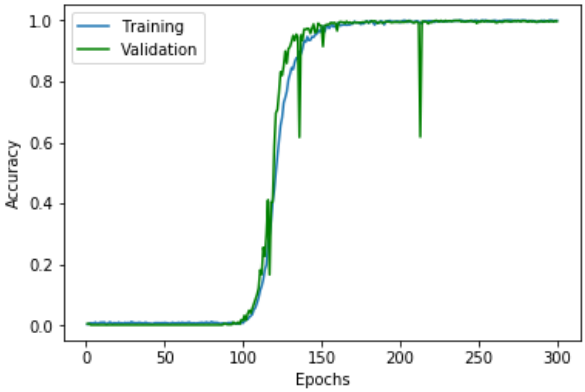
Pada bagian berikut ini akan dimuat hasil uji coba dari 3 skenario pengujian yang telah didefinisikan pada Subbab 5.1.1.

5.2.1 Skenario Pengujian 1: Uji Coba Perbandingan Performa Fungsi Aktivasi ReLU dan LeakyReLU.

Dataset yang digunakan dalam skenario pengujian performa sistem adalah dataset faces94. Gambar 5.1 menunjukkan grafik dari proses *training* CNN ketika fungsi aktivasi yang digunakan adalah ReLU, sedangkan Gambar 5.2 menunjukkan proses *training* menggunakan fungsi aktivasi LeakyReLU.

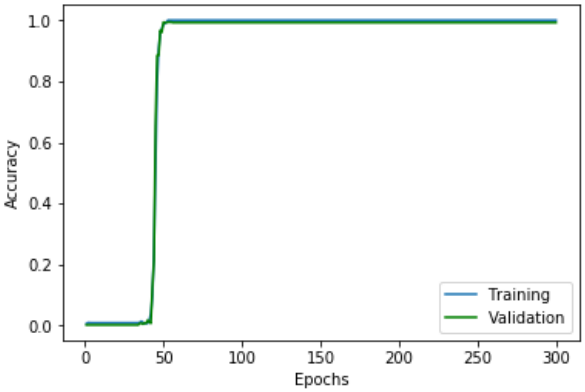
Kedua fungsi aktivasi ini memiliki beberapa perbedaan. Fungsi aktivasi ReLU mengabaikan seluruh nilai negatif dari neuron sebelumnya, sehingga dapat menyebabkan neuron mati jika suatu neuron terus-menerus mendapatkan nilai negatif. Sedangkan fungsi aktivasi LeakyReLU tetap mengambil nilai negatif, namun nilai negatif akan diperkecil. Berikut hasil uji coba dari penggunaan kedua fungsi aktivasi ini:

1. Menggunakan fungsi aktivasi ReLU, grafik proses *training* dapat dilihat pada Gambar 5.1.
   Nilai akurasi *testing*: 99,8%



Gambar 5.1 VGG16 menggunakan ReLU

2. Menggunakan fungsi aktivasi LeakyReLU, grafik proses *training* dapat dilihat pada Gambar 5.2.
   Nilai akurasi *testing*: 99.8%



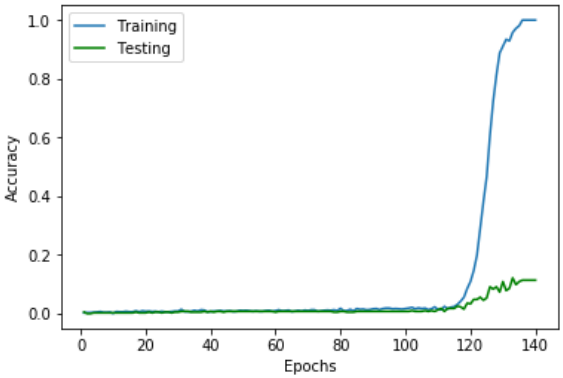
Gambar 5.2 VGG16 menggunakan Leaky ReLU

5.2.2 Skenario Pengujian 2: Uji Coba Perbandingan Variasi Jumlah Citra

Dataset yang digunakan dalam skenario pengujian ini adalah dataset LFW. Pengujian dilakukan untuk mengetahui apakah variasi jumlah citra per kelas berpengaruh terhadap akurasi CNN. Variasi jumlah citra yang digunakan adalah sebagai berikut: dataset LFW variasi A (terdiri dari 10 citra wajah per kelas), dataset LFW variasi B (terdiri dari 40 citra wajah per kelas), dan dataset LFW variasi C (terdiri lebih dari 40 citra wajah per kelas).

1. Dataset LFW Variasi A dan pengujian.

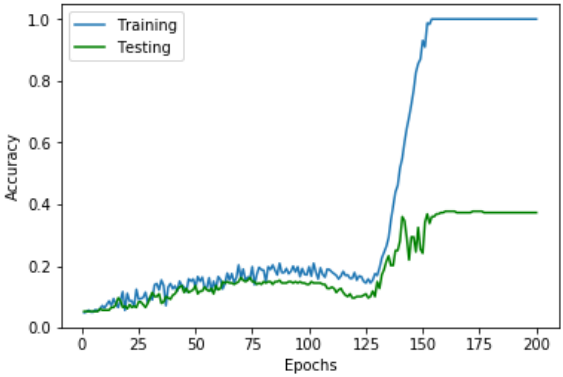
   Pemilihan data dilakukan dengan cara memilih dan mengambil kelas dengan jumlah citra minimal 10. Kemudian dari seluruh kelas yang telah dipilih tersebut, jika jumlah citra diatas 10, cukup ambil 10 citra pertama dan hapus sisanya. Melalui proses pemilihan ini dihasilkan dataset LFW yang konsisten berisi 10 citra per kelasnya dengan jumlah citra total 1.580 (158 kelas). Nilai akurasi *testing* adalah 13,5%. Grafik proses *training* dapat dilihat pada Gambar 5.3.



Gambar 5.3 Hasil uji dataset LFW variasi A

2. Dataset LFW Variasi B dan pengujian.

   Metode pemilihan data dilakukan dengan cara memilih dan mengambil kelas dengan jumlah citra minimal 40. Kemudian dari seluruh kelas yang telah dipilih tersebut, jika jumlah citra diatas 40, cukup ambil 40 citra pertama dan hapus sisanya. Melalui proses pemilihan ini dihasilkan dataset LFW yang konsisten berisi 40 citra per kelasnya. Jumlah citra total pada variasi ini adalah 760 (19 kelas). Walaupun total jumlah citra dan jumlah kelas berkurang, namun dari uji coba diperoleh peningkatan nilai akurasi *testing* menjadi 37,3%. Grafik proses *training* dapat dilihat pada Gambar 5.4.

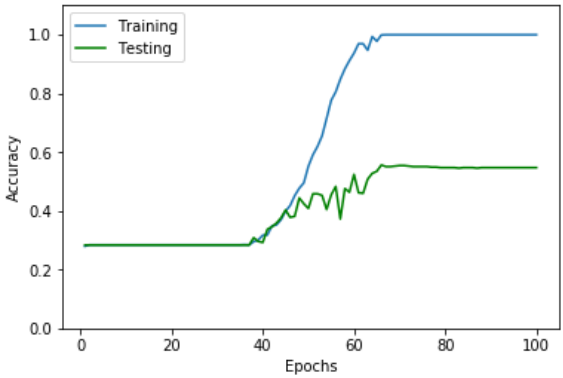


Gambar 5.4 Hasil uji dataset LFW variasi B

3. Dataset LFW Variasi C dan pengujian.

   Untuk menanggulangi total jumlah citra yang sedikit pada uji coba kedua, dipilih dan diambil kelas dengan jumlah citra minimal 40 (untuk mencegah *overfitting*), dan ambil seluruh citra dari kelas tersebut.

Melalui metode pemilihan ini, total jumlah citra ditingkatkan menjadi 1.867 (19 kelas). Dari uji coba diperoleh nilai akurasi *testing* sebesar 54,9%. Grafik proses *training* dapat dilihat pada Gambar 5.5.

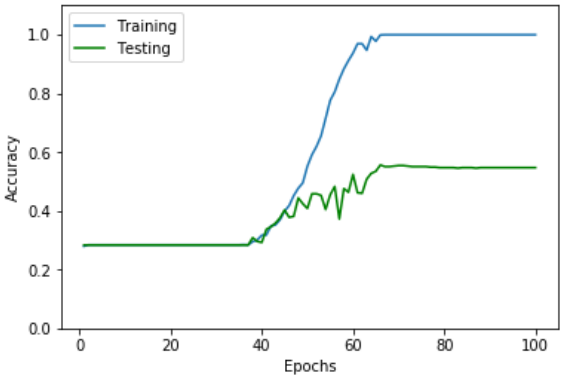


Gambar 5.5 Hasil uji dataset LFW variasi C

5.2.3 Skenario Pengujian 3: Uji Coba Pengaruh Face Alignment dan Augmentasi Dataset Terhadap Akurasi

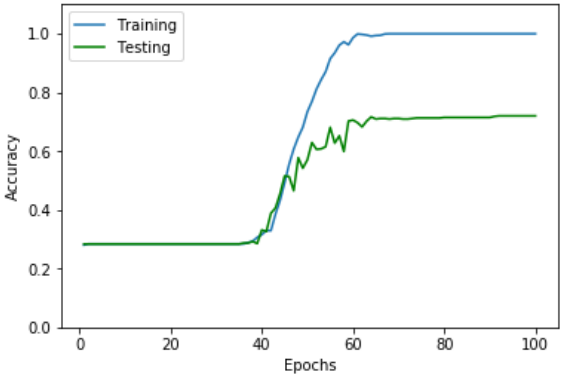
Untuk skenario pengujian ini, akan digunakan dataset LFW variasi C dengan jumlah citra diatas 40 per kelas sebagai *baseline*, kemudian akan diterapkan *face alignment* dan augmentasi data untuk meningkatkan jumlah dataset dengan harapan juga akan terjadi peningkatan akurasi. Metode augmentasi data yang digunakan adalah augmentasi data dengan proses *filtering* dan augmentasi data dengan proses transformasi geometri

1. *Baseline* (Dataset LFW Variasi C)
   Grafik proses *training* dapat dilihat pada Gambar 5.6.
   Nilai akurasi *testing*: 54,9%



Gambar 5.6 Grafik *baseline*

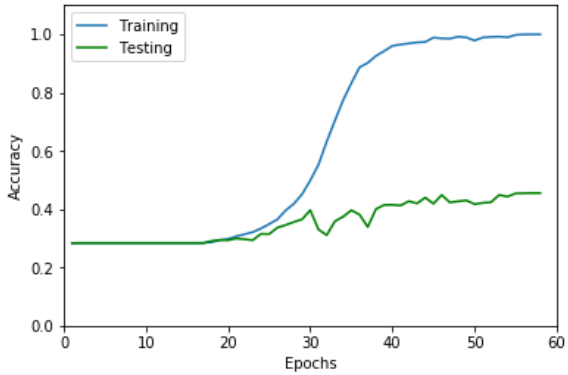
2. Penerapan *face alignment* pada *baseline*
   Grafik proses *training* dapat dilihat pada Gambar 5.7.
   Nilai akurasi *testing*: 72,2%



Gambar 5.7 Grafik uji coba *face alignment*

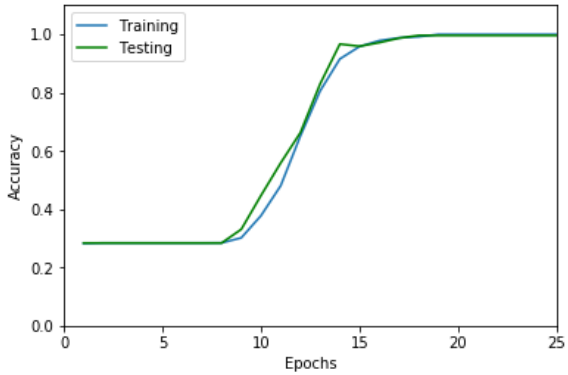
3. Augmentasi data *baseline* dengan proses transformasi geometri.
   Grafik proses *training* dapat dilihat pada Gambar 5.8.

Nilai akurasi *testing*: 45,5%



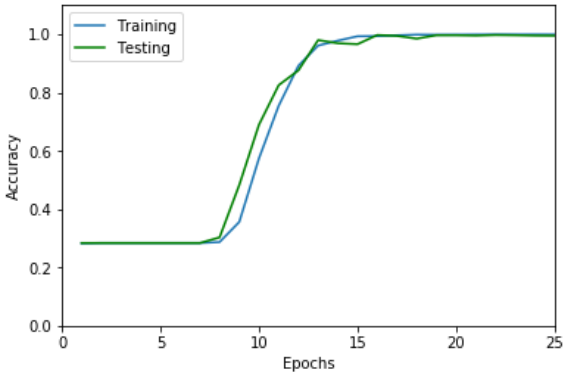
Gambar 5.8 Grafik uji coba augmentasi geometri

4. Augmentasi data *baseline* dengan proses *filtering*.
   Grafik proses *training* dapat dilihat pada Gambar 5.9.
   Nilai akurasi *testing*: 99,6%



Gambar 5.9 Grafik uji coba augmentasi filter

5. Augmentasi data (*baseline* + *face alignment*) dengan proses *filtering*.
   Grafik proses *training* dapat dilihat pada Gambar 5.10.
   Nilai akurasi *testing*: 99,6%



Gambar 5.10 Grafik uji coba augmentasi data dengan proses filtering pada data *baseline + face alignment*

5.3 Analisa dan Evaluasi

Seluruh proses *training* dari hasil uji coba pada Subbab 5.2 sudah mencapai konvergensi, dengan nilai akurasi *training* 100%, yang berbeda adalah nilai akurasi *testing* (kinerja model dalam mengenali wajah), yang akan dianalisa pada subbab ini.

5.3.1 Analisa Skenario Pengujian 1

Berdasarkan uji coba yang dilaksanakan, nilai akurasi *testing* yang diperoleh menggunakan fungsi aktivasi ReLU dan LeakyReLU sama yaitu sebesar 99,8%. Namun kecepatan *training* dari fungsi aktivasi LeakyReLU lebih cepat dibandingkan fungsi aktivasi ReLU. Pada CNN dengan fungsi aktivasi ReLU sistem mencapai konvergensi setelah kurang lebih 200 epoch, seperti yang ditunjukkan oleh Gambar 5.1. Sedangkan dengan fungsi aktivasi LeakyReLU sistem mencapai konvergensi setelah kurang lebih 50 epoch, seperti yang dapat dilihat pada Gambar 5.2.

Skenario uji coba ini menunjukkan LeakyReLU menghasilkan performa yang lebih baik jika dibandingkan dengan fungsi aktivasi ReLU. Hasil ini sesuai dengan penelitian yang dilakukan oleh Bing Xu pada tahun 2015 [11].

5.3.2 Analisa Skenario Pengujian 2

Ketika diambil dataset LFW variasi A, terjadi *overfitting* pada CNN, ini dapat dilihat pada Gambar 5.3. Ketika akurasi *training* sudah maksimal, akurasi *testing* masih bernilai sangat rendah yaitu hanya 13,5%.

Setelah itu diambil dataset LFW variasi B. Akurasi *testing* dari variasi dataset ini adalah 37,3%. Dari Gambar 5.4, dapat diamati walaupun terhitung masih buruk, namun akurasi *testing* dari CNN sudah meningkat dibandingkan uji coba dengan 10 jumlah citra/kelas.

Kemudian dilakukan pengujian pada dataset LFW variasi C dan dari pengujian ini nilai akurasi *testing* sistem meningkat menjadi 54.9%.

Melalui skenario pengujian ini, diamati bahwa penambahan jumlah citra yang terdapat dalam 1 kelas meningkatkan akurasi dari sistem. Ini dikarenakan semakin banyak citra dalam satu kelas, semakin rendah kemungkinan CNN untuk menghasilkan model yang *overfitting*.

5.3.3 Analisa Skenario Pengujian 3

Dari skenario pengujian 3, dapat diamati bahwa *face alignment* meningkatkan akurasi sistem dari nilai akurasi awal 54,9% menjadi 72,1%. Augmentasi data menggunakan metode *filtering* meningkatkan akurasi sistem dari nilai akurasi awal 54,9% menjadi 99,6%, sedangkan augmentasi data menggunakan metode transformasi geometri menurunkan akurasi dari sistem dari nilai akurasi awal 54,9% menjadi 45,5%. Menerapkan *face alignment* dan augmentasi data

(dengan proses *filtering*), tidak menghasilkan peningkatan nilai akurasi jika dibandingkan dengan augmentasi data (dengan proses *filtering*) saja, namun penelitian lebih lanjut masih diperlukan.

5.3.4 Evaluasi Model

Model akhir yang diperoleh adalah model CNN dengan nilai akurasi maksimal dari seluruh uji coba yang dilakukan (99,6%), yang dicapai menggunakan:

- Dataset LFW Variasi 3 dengan jumlah citra melebihi 40 per kelas.
- Menerapkan face alignment.
- Melakukan augmentasi data menggunakan proses *filtering*

Nilai akurasi 99,6% menandakan ada kesalahan prediksi pada model. Dari 2.801 *test data*, terdapat 10 kesalahan prediksi. Hasil prediksi salah ini akan dimuat pada Tabel 5.1. Dari tabel, diketahui bahwa 10 citra yang salah diprediksi tersebut merupakan kumpulan citra dari 2 orang, yaitu citra wajah dari Colin Powel yang salah diprediksi sebagai Hugo Chavez dan citra wajah dari Donald Rumsfeld yang salah diprediksi sebagai George W. Bush.

Tabel 5.2 Hasil Prediksi Salah

ID	Prediksi	Sebenarnya
67	 Hugo Chavez	 Colin Powell

146	 George W Bush	 Donald Rumsfeld
183	 Hugo Chavez	 Colin Powell
306	 George W Bush	 Donald Rumsfeld
350	 George W Bush	 Donald Rumsfeld

601	 George W Bush	 Donald Rumsfeld
683	 Hugo Chavez	 Colin Powell
1296	 George W Bush	 Donald Rumsfeld
1768	 Hugo Chavez	 Colin Powell

2240		
	Hugo Chavez	Colin Powell

Citra wajah Donald Rumsfeld memang mirip dengan George W Bush sehingga terjadi kesalahan prediksi. Namun citra wajah dari Colin Powell tidak mirip dengan citra wajah Hugo Chavez. Kesalahan yang terjadi mungkin disebabkan karena citra wajah Colin Powell yang diambil merupakan outlier dari kluster citra wajah Colin Powell lainnya. Hasil prediksi selengkapnya dapat dilihat pada bagian Lampiran.

BAB VI
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Beberapa kesimpulan yang dapat diambil dari tugas akhir ini adalah:

1. Fungsi aktivasi Leaky ReLU lebih cepat dibandingkan fungsi aktivasi ReLU, setelah dilakukan pengujian menggunakan dataset faces94. Pada fungsi aktivasi ReLU, dibutuhkan kurang lebih 200 epoch untuk mencapai konvergensi sedangkan pada fungsi aktivasi Leaky ReLU, hanya dibutuhkan kurang lebih 50 epoch.
2. Dari uji coba variasi jumlah dataset LFW yang digunakan, dataset LFW Variasi C yang memiliki jumlah citra per kelas terbanyak memiliki nilai akurasi *testing* tertinggi (sebesar 54,9%). Dapat ditarik kesimpulan bahwa semakin banyak jumlah citra per kelas, maka semakin tinggi kinerja dari CNN.
3. Praproses face alignment meningkatkan nilai akurasi sistem dari yang awalnya 54,9% menjadi 72,2% atau sebanyak 17,3%.
4. Augmentasi data menggunakan proses *filtering* pada dataset LFW meningkatkan nilai akurasi sistem dari yang awalnya 54,9% menjadi 99,6%, sedangkan augmentasi data menggunakan proses transformasi geometri tidak meningkatkan kinerja sistem, bahkan menurunkan nilai akurasi sistem dari yang awalnya 54,9% menjadi 45,5%.
5. Nilai akurasi maksimal 99,6% diperoleh pada dataset LFW Variasi C dengan jumlah citra lebih dari 40 per kelas, setelah face alignment dan augmentasi data (menggunakan proses *filtering*) dilakukan.

6.2 Saran

Beberapa hal yang masih dapat ditingkatkan dari penelitian ini adalah sebagai berikut:

1. Masih dapat dilakukan penelitian lebih lanjut mengenai efek penggabungan augmentasi filter dan face alignment.
2. Diperlukan data yang jauh lebih banyak untuk menghasilkan model yang lebih "cerdas". Dalam penelitian ini hanya menggunakan kurang dari 10.000 data, sedangkan pada paper referensi menggunakan 2 juta dataset gabungan. Data paper referensi yang digunakan hanya LFW, karena dataset lainnya tidak tersedia untuk publik.
3. Masih dapat dilakukan penelitian lebih lanjut mengenai efek resolusi gambar terhadap akurasi dan harga komputasi CNN.

DAFTAR PUSTAKA

[1] J. Aron, "Google DeepMind AI outplays humans at video games," NewScientist, [Online]. Available: https://www.newscientist.com/article/dn27031-google-deepmind-ai-outplays-humans-at-video-games/. [Accessed 6 Juli 2018].

[2] University of Massachusetts, "Labeled Faces in the Wild," [Online]. Available: http://vis-www.cs.umass.edu/lfw/. [Accessed 22 Desember 2017].

[3] University of Essex, "Collection of Facial Images: Faces94," [Online]. Available: http://cswww.essex.ac.uk/mv/allfaces/faces94.html. [Accessed 29 Mei 2018].

[4] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud and V. Shet, Multi-digit Number Recognition from Street View Imagery Using Deep Convolutional Neural Networks, 2014.

[5] CS231, "CS231 Course," [Online]. Available: http://cs231n.github.io/convolutional-networks/. [Accessed 22 Desember 2017].

[6] D. Britz, "Understanding Convolutional Neural Networks for NLP," [Online]. Available: http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/. [Accessed 28 Desember 2017].

[7] P. Veličković, "Deep learning for complete beginners: convolutional neural networks with keras," [Online]. Available: https://cambridgespark.com/content/tutorials/convolutional-neural-networks-with-keras/index.html. [Accessed 1 April 2018].

[8] D. Gupta, "Architecture of Convolutional Neural Networks (CNN) demystified," [Online]. Available: https://www.analyticsvidhya.com/blog/2017/06/architectu

re-of-convolutional-neural-networks-simplified-demystified/. [Accessed 29 Mei 2018].

[9] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Machine Learning Research*, no. 15, 2014.

[10] deeplearning.ai, "Why do you need non-linear activation functions?," [Online]. Available: https://www.coursera.org/learn/neural-networks-deep-learning/lecture/OASKH/why-do-you-need-non-linear-activation-functions. [Accessed 5 Juni 2018].

[11] B. Xu, N. Wang, T. Chen and M. Li, "Empirical Evaluation of Rectified Activations in Convolution Network," in *ICML Deep Learning Workshop*, 2015.

[12] A. S. Walia, "Activation functions and its types - Which is better?," [Online]. Available: https://towardsdatascience.com/activation-functions-and-its-types-which-is-better-a9a5310cc8f. [Accessed 1 Mei 2018].

[13] A. Geitgey, "Machine Learning is Fun! Part 4," [Online]. Available: https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78. [Accessed 22 Desember 2017].

[14] V. Kazemi and J. Sullivan, One Millisecond Face Alignment With an Ensemble of Regression Trees, Sweden, 2014.

[15] D. E. King, "Dlib-ml: A Machine Learning Toolkit," *Journal of Machine Learning Research*, no. 10, pp. 1755-1758, 2009.

[16] B. Amos, B. Ludwiczuk and M. Satyanarayanan, "Openface: A general-purpose face recognition library with mobile applications," CMU School of Computer Science, Tech. Rep., 2016.

[17] O. Parkhi, A. Vedaldi and A. Zisserman, Deep Face Recognition, 2015.

[18] Python, "Python," [Online]. Available: https://www.python.org/. [Accessed 22 Mei 2018].

[19] Tensorflow Team, "Tensorflow," [Online]. Available: https://www.tensorflow.org/. [Accessed 15 Mei 2018].

[20] F. Chollet and others, "Keras," Keras, 2015.

[21] A. Baratloo, M. Hosseini, A. Negida and G. El Ashal, "Part 1: Simple Definition and Calculation of Accuracy, Sensitivity and Specificity," *Emergency 3.2*, pp. 48-49, 2015.

[22] Keras, "Usage of Metrics," Keras, [Online]. Available: https://keras.io/metrics/. [Accessed 5 Juli 2018].

[23] NVIDIA, "NVIDIA cuDNN," [Online]. Available: https://developer.nvidia.com/cudnn. [Accessed 1 April 2018].

[24] F. Shaikh, "Why are GPUs necessary for training Deep Learning models?," Analytics Vidhya, [Online]. Available: https://www.analyticsvidhya.com/blog/2017/05/gpus-necessary-for-deep-learning/. [Accessed 25 Juni 2018].

[25] J. Jordan, "Hyperparameter tuning for machine learning models.," [Online]. Available: https://www.jeremyjordan.me/hyperparameter-tuning/. [Accessed 15 Mei 2018].

[26] Z. Reitermanov´a, "Data Splitting," in *WDS'10 Proceedings of Contributed Papers*, 2010.

[27] Scikit-Learn, "Stratified Shuffle Split," [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html. [Accessed 28 Juni 2018].

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Vincent Daniel Win (lahir 22 Januari 1997, umur 21 ketika penulisan, Makassar, Sulawesi Selatan, Indonesia). Menempuh sekolah dasar di SD IPEKA Makassar, menengah pertama di SMP Zion Makassar, dan menengah atas di SMA Zion Makassar. Pada tahun 2014 diterima di ITS Surabaya melalui jalur SBMPTN. Pernah menjadi staff departmen Ristek HMTC selama satu periode, menjadi Ketua Panitia 3C 2016, dan menjadi Kepala Departemen dari organisasi KMK ITS periode 2016-2017.

Email : vincent.daniel14@mhs.if.its.ac.id

[Halaman ini sengaja dikosongkan]

LAMPIRAN

Hasil Prediksi

Pada tabel berikut akan ditampilkan 500 hasil prediksi pertama dari 2.801 test data. Citra wajah dari 500 hasil prediksi ini terlampir dibawah tabel.
Hasil prediksi yang salah telah dikelompokkan pada Subbab 5.3.4 Evaluasi Model (hanya terdapat 10 kesalahan prediksi).

ID	Prediksi	Sebenarnya
1	Junichiro_Koizumi	Junichiro_Koizumi
2	Serena_Williams	Serena_Williams
3	George_W_Bush	George_W_Bush
4	Donald_Rumsfeld	Donald_Rumsfeld
5	Tony_Blair	Tony_Blair
6	Vladimir_Putin	Vladimir_Putin
7	George_W_Bush	George_W_Bush
8	Tony_Blair	Tony_Blair
9	George_W_Bush	George_W_Bush
10	Ariel_Sharon	Ariel_Sharon
11	George_W_Bush	George_W_Bush
12	George_W_Bush	George_W_Bush
13	Arnold_Schwarzenegger	Arnold_Schwarzenegger
14	Colin_Powell	Colin_Powell
15	Donald_Rumsfeld	Donald_Rumsfeld
16	Donald_Rumsfeld	Donald_Rumsfeld
17	Junichiro_Koizumi	Junichiro_Koizumi
18	Tony_Blair	Tony_Blair
19	Junichiro_Koizumi	Junichiro_Koizumi
20	George_W_Bush	George_W_Bush

21	Colin_Powell	Colin_Powell
22	George_W_Bush	George_W_Bush
23	Jennifer_Capriati	Jennifer_Capriati
24	Hugo_Chavez	Hugo_Chavez
25	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
26	George_W_Bush	George_W_Bush
27	Gerhard_Schroeder	Gerhard_Schroeder
28	Colin_Powell	Colin_Powell
29	Jean_Chretien	Jean_Chretien
30	Serena_Williams	Serena_Williams
31	George_W_Bush	George_W_Bush
32	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
33	George_W_Bush	George_W_Bush
34	Donald_Rumsfeld	Donald_Rumsfeld
35	Lleyton_Hewitt	Lleyton_Hewitt
36	Hugo_Chavez	Hugo_Chavez
37	George_W_Bush	George_W_Bush
38	John_Ashcroft	John_Ashcroft
39	Tony_Blair	Tony_Blair
40	Vladimir_Putin	Vladimir_Putin
41	Lleyton_Hewitt	Lleyton_Hewitt
42	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
43	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
44	Jean_Chretien	Jean_Chretien
45	Colin_Powell	Colin_Powell
46	George_W_Bush	George_W_Bush
47	Ariel_Sharon	Ariel_Sharon
48	Serena_Williams	Serena_Williams
49	George_W_Bush	George_W_Bush

50	Donald_Rumsfeld	Donald_Rumsfeld
51	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
52	Colin_Powell	Colin_Powell
53	Junichiro_Koizumi	Junichiro_Koizumi
54	George_W_Bush	George_W_Bush
55	John_Ashcroft	John_Ashcroft
56	George_W_Bush	George_W_Bush
57	Vladimir_Putin	Vladimir_Putin
58	Serena_Williams	Serena_Williams
59	Gerhard_Schroeder	Gerhard_Schroeder
60	Colin_Powell	Colin_Powell
61	Donald_Rumsfeld	Donald_Rumsfeld
62	Colin_Powell	Colin_Powell
63	Donald_Rumsfeld	Donald_Rumsfeld
64	Tony_Blair	Tony_Blair
65	Gerhard_Schroeder	Gerhard_Schroeder
66	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
67	Hugo_Chavez	Colin_Powell
68	George_W_Bush	George_W_Bush
69	George_W_Bush	George_W_Bush
70	Tony_Blair	Tony_Blair
71	Gerhard_Schroeder	Gerhard_Schroeder
72	Gerhard_Schroeder	Gerhard_Schroeder
73	George_W_Bush	George_W_Bush
74	Tony_Blair	Tony_Blair
75	Gerhard_Schroeder	Gerhard_Schroeder
76	George_W_Bush	George_W_Bush
77	Tony_Blair	Tony_Blair
78	Jacques_Chirac	Jacques_Chirac

79	George_W_Bush	George_W_Bush
80	Jacques_Chirac	Jacques_Chirac
81	Ariel_Sharon	Ariel_Sharon
82	George_W_Bush	George_W_Bush
83	Colin_Powell	Colin_Powell
84	George_W_Bush	George_W_Bush
85	George_W_Bush	George_W_Bush
86	Arnold_Schwarzenegger	Arnold_Schwarzenegger
87	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
88	George_W_Bush	George_W_Bush
89	Tony_Blair	Tony_Blair
90	Jean_Chretien	Jean_Chretien
91	Laura_Bush	Laura_Bush
92	Jennifer_Capriati	Jennifer_Capriati
93	Junichiro_Koizumi	Junichiro_Koizumi
94	George_W_Bush	George_W_Bush
95	George_W_Bush	George_W_Bush
96	Colin_Powell	Colin_Powell
97	George_W_Bush	George_W_Bush
98	George_W_Bush	George_W_Bush
99	Jean_Chretien	Jean_Chretien
100	Hugo_Chavez	Hugo_Chavez
101	Donald_Rumsfeld	Donald_Rumsfeld
102	Serena_Williams	Serena_Williams
103	Jean_Chretien	Jean_Chretien
104	Tony_Blair	Tony_Blair
105	Vladimir_Putin	Vladimir_Putin
106	George_W_Bush	George_W_Bush
107	George_W_Bush	George_W_Bush

108	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
109	Colin_Powell	Colin_Powell
110	Tony_Blair	Tony_Blair
111	Jean_Chretien	Jean_Chretien
112	Colin_Powell	Colin_Powell
113	Colin_Powell	Colin_Powell
114	John_Ashcroft	John_Ashcroft
115	Junichiro_Koizumi	Junichiro_Koizumi
116	Donald_Rumsfeld	Donald_Rumsfeld
117	Jacques_Chirac	Jacques_Chirac
118	Serena_Williams	Serena_Williams
119	Colin_Powell	Colin_Powell
120	Colin_Powell	Colin_Powell
121	George_W_Bush	George_W_Bush
122	Vladimir_Putin	Vladimir_Putin
123	Donald_Rumsfeld	Donald_Rumsfeld
124	John_Ashcroft	John_Ashcroft
125	Tony_Blair	Tony_Blair
126	Colin_Powell	Colin_Powell
127	Vladimir_Putin	Vladimir_Putin
128	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
129	George_W_Bush	George_W_Bush
130	George_W_Bush	George_W_Bush
131	Gerhard_Schroeder	Gerhard_Schroeder
132	Tony_Blair	Tony_Blair
133	Jacques_Chirac	Jacques_Chirac
134	George_W_Bush	George_W_Bush
135	Jennifer_Capriati	Jennifer_Capriati
136	Hugo_Chavez	Hugo_Chavez

137	Jennifer_Capriati	Jennifer_Capriati
138	Colin_Powell	Colin_Powell
139	Junichiro_Koizumi	Junichiro_Koizumi
140	Lleyton_Hewitt	Lleyton_Hewitt
141	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
142	Gerhard_Schroeder	Gerhard_Schroeder
143	Colin_Powell	Colin_Powell
144	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
145	George_W_Bush	George_W_Bush
146	George_W_Bush	Donald_Rumsfeld
147	Ariel_Sharon	Ariel_Sharon
148	George_W_Bush	George_W_Bush
149	Ariel_Sharon	Ariel_Sharon
150	Laura_Bush	Laura_Bush
151	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
152	George_W_Bush	George_W_Bush
153	Hugo_Chavez	Hugo_Chavez
154	Lleyton_Hewitt	Lleyton_Hewitt
155	Serena_Williams	Serena_Williams
156	George_W_Bush	George_W_Bush
157	George_W_Bush	George_W_Bush
158	Colin_Powell	Colin_Powell
159	Colin_Powell	Colin_Powell
160	George_W_Bush	George_W_Bush
161	Gerhard_Schroeder	Gerhard_Schroeder
162	Ariel_Sharon	Ariel_Sharon
163	George_W_Bush	George_W_Bush
164	Colin_Powell	Colin_Powell
165	Junichiro_Koizumi	Junichiro_Koizumi

166	Tony_Blair	Tony_Blair
167	Ariel_Sharon	Ariel_Sharon
168	George_W_Bush	George_W_Bush
169	Gerhard_Schroeder	Gerhard_Schroeder
170	Serena_Williams	Serena_Williams
171	Jennifer_Capriati	Jennifer_Capriati
172	Jennifer_Capriati	Jennifer_Capriati
173	George_W_Bush	George_W_Bush
174	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
175	Gerhard_Schroeder	Gerhard_Schroeder
176	George_W_Bush	George_W_Bush
177	George_W_Bush	George_W_Bush
178	Junichiro_Koizumi	Junichiro_Koizumi
179	George_W_Bush	George_W_Bush
180	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
181	George_W_Bush	George_W_Bush
182	George_W_Bush	George_W_Bush
183	Hugo_Chavez	Colin_Powell
184	Lleyton_Hewitt	Lleyton_Hewitt
185	George_W_Bush	George_W_Bush
186	Donald_Rumsfeld	Donald_Rumsfeld
187	Tony_Blair	Tony_Blair
188	Arnold_Schwarzenegger	Arnold_Schwarzenegger
189	Colin_Powell	Colin_Powell
190	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
191	George_W_Bush	George_W_Bush
192	Colin_Powell	Colin_Powell
193	George_W_Bush	George_W_Bush
194	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva

195	John_Ashcroft	John_Ashcroft
196	George_W_Bush	George_W_Bush
197	Ariel_Sharon	Ariel_Sharon
198	Laura_Bush	Laura_Bush
199	George_W_Bush	George_W_Bush
200	George_W_Bush	George_W_Bush
201	Ariel_Sharon	Ariel_Sharon
202	Jacques_Chirac	Jacques_Chirac
203	Colin_Powell	Colin_Powell
204	George_W_Bush	George_W_Bush
205	Tony_Blair	Tony_Blair
206	George_W_Bush	George_W_Bush
207	Gerhard_Schroeder	Gerhard_Schroeder
208	Lleyton_Hewitt	Lleyton_Hewitt
209	Jean_Chretien	Jean_Chretien
210	Colin_Powell	Colin_Powell
211	George_W_Bush	George_W_Bush
212	George_W_Bush	George_W_Bush
213	George_W_Bush	George_W_Bush
214	Jacques_Chirac	Jacques_Chirac
215	Donald_Rumsfeld	Donald_Rumsfeld
216	John_Ashcroft	John_Ashcroft
217	George_W_Bush	George_W_Bush
218	George_W_Bush	George_W_Bush
219	Colin_Powell	Colin_Powell
220	Colin_Powell	Colin_Powell
221	Serena_Williams	Serena_Williams
222	Colin_Powell	Colin_Powell
223	Donald_Rumsfeld	Donald_Rumsfeld

224	Gerhard_Schroeder	Gerhard_Schroeder
225	Vladimir_Putin	Vladimir_Putin
226	Colin_Powell	Colin_Powell
227	Tony_Blair	Tony_Blair
228	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
229	Colin_Powell	Colin_Powell
230	Colin_Powell	Colin_Powell
231	Vladimir_Putin	Vladimir_Putin
232	Jacques_Chirac	Jacques_Chirac
233	George_W_Bush	George_W_Bush
234	Colin_Powell	Colin_Powell
235	George_W_Bush	George_W_Bush
236	John_Ashcroft	John_Ashcroft
237	Colin_Powell	Colin_Powell
238	Gerhard_Schroeder	Gerhard_Schroeder
239	Gerhard_Schroeder	Gerhard_Schroeder
240	John_Ashcroft	John_Ashcroft
241	Colin_Powell	Colin_Powell
242	Colin_Powell	Colin_Powell
243	Gerhard_Schroeder	Gerhard_Schroeder
244	Colin_Powell	Colin_Powell
245	George_W_Bush	George_W_Bush
246	George_W_Bush	George_W_Bush
247	Colin_Powell	Colin_Powell
248	George_W_Bush	George_W_Bush
249	Colin_Powell	Colin_Powell
250	George_W_Bush	George_W_Bush
251	George_W_Bush	George_W_Bush
252	Vladimir_Putin	Vladimir_Putin

253	Serena_Williams	Serena_Williams
254	George_W_Bush	George_W_Bush
255	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
256	Arnold_Schwarzenegger	Arnold_Schwarzenegger
257	Lleyton_Hewitt	Lleyton_Hewitt
258	George_W_Bush	George_W_Bush
259	George_W_Bush	George_W_Bush
260	Gerhard_Schroeder	Gerhard_Schroeder
261	George_W_Bush	George_W_Bush
262	Arnold_Schwarzenegger	Arnold_Schwarzenegger
263	Hugo_Chavez	Hugo_Chavez
264	George_W_Bush	George_W_Bush
265	George_W_Bush	George_W_Bush
266	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
267	George_W_Bush	George_W_Bush
268	Ariel_Sharon	Ariel_Sharon
269	Colin_Powell	Colin_Powell
270	Colin_Powell	Colin_Powell
271	Tony_Blair	Tony_Blair
272	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
273	Vladimir_Putin	Vladimir_Putin
274	Junichiro_Koizumi	Junichiro_Koizumi
275	Jacques_Chirac	Jacques_Chirac
276	John_Ashcroft	John_Ashcroft
277	Junichiro_Koizumi	Junichiro_Koizumi
278	George_W_Bush	George_W_Bush
279	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
280	Serena_Williams	Serena_Williams
281	Ariel_Sharon	Ariel_Sharon

282	Colin_Powell	Colin_Powell
283	Tony_Blair	Tony_Blair
284	Tony_Blair	Tony_Blair
285	Arnold_Schwarzenegger	Arnold_Schwarzenegger
286	Colin_Powell	Colin_Powell
287	Jean_Chretien	Jean_Chretien
288	Junichiro_Koizumi	Junichiro_Koizumi
289	Lleyton_Hewitt	Lleyton_Hewitt
290	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
291	Colin_Powell	Colin_Powell
292	George_W_Bush	George_W_Bush
293	Gerhard_Schroeder	Gerhard_Schroeder
294	Gerhard_Schroeder	Gerhard_Schroeder
295	Donald_Rumsfeld	Donald_Rumsfeld
296	Vladimir_Putin	Vladimir_Putin
297	George_W_Bush	George_W_Bush
298	Tony_Blair	Tony_Blair
299	Ariel_Sharon	Ariel_Sharon
300	Jean_Chretien	Jean_Chretien
301	Ariel_Sharon	Ariel_Sharon
302	George_W_Bush	George_W_Bush
303	George_W_Bush	George_W_Bush
304	Tony_Blair	Tony_Blair
305	Laura_Bush	Laura_Bush
306	George_W_Bush	Donald_Rumsfeld
307	Laura_Bush	Laura_Bush
308	Colin_Powell	Colin_Powell
309	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
310	Gerhard_Schroeder	Gerhard_Schroeder

311	George_W_Bush	George_W_Bush
312	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
313	George_W_Bush	George_W_Bush
314	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
315	Junichiro_Koizumi	Junichiro_Koizumi
316	Donald_Rumsfeld	Donald_Rumsfeld
317	Jean_Chretien	Jean_Chretien
318	Colin_Powell	Colin_Powell
319	George_W_Bush	George_W_Bush
320	George_W_Bush	George_W_Bush
321	Gerhard_Schroeder	Gerhard_Schroeder
322	Vladimir_Putin	Vladimir_Putin
323	George_W_Bush	George_W_Bush
324	George_W_Bush	George_W_Bush
325	Gerhard_Schroeder	Gerhard_Schroeder
326	George_W_Bush	George_W_Bush
327	Tony_Blair	Tony_Blair
328	George_W_Bush	George_W_Bush
329	George_W_Bush	George_W_Bush
330	Gerhard_Schroeder	Gerhard_Schroeder
331	George_W_Bush	George_W_Bush
332	Jennifer_Capriati	Jennifer_Capriati
333	George_W_Bush	George_W_Bush
334	Jacques_Chirac	Jacques_Chirac
335	George_W_Bush	George_W_Bush
336	Colin_Powell	Colin_Powell
337	George_W_Bush	George_W_Bush
338	George_W_Bush	George_W_Bush
339	Arnold_Schwarzenegger	Arnold_Schwarzenegger

340	Colin_Powell	Colin_Powell
341	Laura_Bush	Laura_Bush
342	George_W_Bush	George_W_Bush
343	George_W_Bush	George_W_Bush
344	George_W_Bush	George_W_Bush
345	George_W_Bush	George_W_Bush
346	Gerhard_Schroeder	Gerhard_Schroeder
347	Donald_Rumsfeld	Donald_Rumsfeld
348	Colin_Powell	Colin_Powell
349	John_Ashcroft	John_Ashcroft
350	George_W_Bush	Donald_Rumsfeld
351	Laura_Bush	Laura_Bush
352	Hugo_Chavez	Hugo_Chavez
353	Colin_Powell	Colin_Powell
354	George_W_Bush	George_W_Bush
355	Gerhard_Schroeder	Gerhard_Schroeder
356	Gerhard_Schroeder	Gerhard_Schroeder
357	George_W_Bush	George_W_Bush
358	George_W_Bush	George_W_Bush
359	Colin_Powell	Colin_Powell
360	Colin_Powell	Colin_Powell
361	Tony_Blair	Tony_Blair
362	Gerhard_Schroeder	Gerhard_Schroeder
363	Colin_Powell	Colin_Powell
364	Jean_Chretien	Jean_Chretien
365	Jacques_Chirac	Jacques_Chirac
366	John_Ashcroft	John_Ashcroft
367	Tony_Blair	Tony_Blair
368	Gerhard_Schroeder	Gerhard_Schroeder

369	George_W_Bush	George_W_Bush
370	Donald_Rumsfeld	Donald_Rumsfeld
371	George_W_Bush	George_W_Bush
372	Arnold_Schwarzenegger	Arnold_Schwarzenegger
373	Ariel_Sharon	Ariel_Sharon
374	Hugo_Chavez	Hugo_Chavez
375	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
376	Colin_Powell	Colin_Powell
377	George_W_Bush	George_W_Bush
378	Tony_Blair	Tony_Blair
379	Hugo_Chavez	Hugo_Chavez
380	Jean_Chretien	Jean_Chretien
381	Vladimir_Putin	Vladimir_Putin
382	Lleyton_Hewitt	Lleyton_Hewitt
383	George_W_Bush	George_W_Bush
384	Colin_Powell	Colin_Powell
385	Ariel_Sharon	Ariel_Sharon
386	John_Ashcroft	John_Ashcroft
387	Hugo_Chavez	Hugo_Chavez
388	Hugo_Chavez	Hugo_Chavez
389	Vladimir_Putin	Vladimir_Putin
390	Ariel_Sharon	Ariel_Sharon
391	George_W_Bush	George_W_Bush
392	Donald_Rumsfeld	Donald_Rumsfeld
393	Ariel_Sharon	Ariel_Sharon
394	George_W_Bush	George_W_Bush
395	Jacques_Chirac	Jacques_Chirac
396	Colin_Powell	Colin_Powell
397	George_W_Bush	George_W_Bush

398	Jennifer_Capriati	Jennifer_Capriati
399	Gerhard_Schroeder	Gerhard_Schroeder
400	Gerhard_Schroeder	Gerhard_Schroeder
401	Tony_Blair	Tony_Blair
402	George_W_Bush	George_W_Bush
403	Serena_Williams	Serena_Williams
404	Tony_Blair	Tony_Blair
405	Colin_Powell	Colin_Powell
406	Tony_Blair	Tony_Blair
407	Hugo_Chavez	Hugo_Chavez
408	George_W_Bush	George_W_Bush
409	Jacques_Chirac	Jacques_Chirac
410	Ariel_Sharon	Ariel_Sharon
411	Tony_Blair	Tony_Blair
412	Serena_Williams	Serena_Williams
413	Donald_Rumsfeld	Donald_Rumsfeld
414	Jean_Chretien	Jean_Chretien
415	George_W_Bush	George_W_Bush
416	Donald_Rumsfeld	Donald_Rumsfeld
417	Colin_Powell	Colin_Powell
418	Junichiro_Koizumi	Junichiro_Koizumi
419	Junichiro_Koizumi	Junichiro_Koizumi
420	Tony_Blair	Tony_Blair
421	Jacques_Chirac	Jacques_Chirac
422	George_W_Bush	George_W_Bush
423	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
424	Arnold_Schwarzenegger	Arnold_Schwarzenegger
425	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
426	George_W_Bush	George_W_Bush

427	George_W_Bush	George_W_Bush
428	Jennifer_Capriati	Jennifer_Capriati
429	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
430	Arnold_Schwarzenegger	Arnold_Schwarzenegger
431	Ariel_Sharon	Ariel_Sharon
432	John_Ashcroft	John_Ashcroft
433	Donald_Rumsfeld	Donald_Rumsfeld
434	Arnold_Schwarzenegger	Arnold_Schwarzenegger
435	Tony_Blair	Tony_Blair
436	Colin_Powell	Colin_Powell
437	Colin_Powell	Colin_Powell
438	Jacques_Chirac	Jacques_Chirac
439	George_W_Bush	George_W_Bush
440	Gerhard_Schroeder	Gerhard_Schroeder
441	George_W_Bush	George_W_Bush
442	Arnold_Schwarzenegger	Arnold_Schwarzenegger
443	George_W_Bush	George_W_Bush
444	Colin_Powell	Colin_Powell
445	Hugo_Chavez	Hugo_Chavez
446	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
447	Hugo_Chavez	Hugo_Chavez
448	Jean_Chretien	Jean_Chretien
449	Luiz_Inacio_Lula_da_Silva	Luiz_Inacio_Lula_da_Silva
450	George_W_Bush	George_W_Bush
451	George_W_Bush	George_W_Bush
452	Gerhard_Schroeder	Gerhard_Schroeder
453	Donald_Rumsfeld	Donald_Rumsfeld
454	Ariel_Sharon	Ariel_Sharon
455	Gerhard_Schroeder	Gerhard_Schroeder

456	Tony_Blair	Tony_Blair
457	George_W_Bush	George_W_Bush
458	George_W_Bush	George_W_Bush
459	Vladimir_Putin	Vladimir_Putin
460	George_W_Bush	George_W_Bush
461	Ariel_Sharon	Ariel_Sharon
462	Ariel_Sharon	Ariel_Sharon
463	Gerhard_Schroeder	Gerhard_Schroeder
464	Ariel_Sharon	Ariel_Sharon
465	Colin_Powell	Colin_Powell
466	George_W_Bush	George_W_Bush
467	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
468	Tony_Blair	Tony_Blair
469	George_W_Bush	George_W_Bush
470	Junichiro_Koizumi	Junichiro_Koizumi
471	Arnold_Schwarzenegger	Arnold_Schwarzenegger
472	Serena_Williams	Serena_Williams
473	Hugo_Chavez	Hugo_Chavez
474	Tony_Blair	Tony_Blair
475	Junichiro_Koizumi	Junichiro_Koizumi
476	Serena_Williams	Serena_Williams
477	George_W_Bush	George_W_Bush
478	George_W_Bush	George_W_Bush
479	George_W_Bush	George_W_Bush
480	John_Ashcroft	John_Ashcroft
481	Laura_Bush	Laura_Bush
482	Hugo_Chavez	Hugo_Chavez
483	Donald_Rumsfeld	Donald_Rumsfeld
484	George_W_Bush	George_W_Bush

485	Donald_Rumsfeld	Donald_Rumsfeld
486	Gloria_Macapagal_Arroyo	Gloria_Macapagal_Arroyo
487	Colin_Powell	Colin_Powell
488	Lleyton_Hewitt	Lleyton_Hewitt
489	Donald_Rumsfeld	Donald_Rumsfeld
490	George_W_Bush	George_W_Bush
491	Tony_Blair	Tony_Blair
492	George_W_Bush	George_W_Bush
493	Jacques_Chirac	Jacques_Chirac
494	George_W_Bush	George_W_Bush
495	Jennifer_Capriati	Jennifer_Capriati
496	George_W_Bush	George_W_Bush
497	Colin_Powell	Colin_Powell
498	Colin_Powell	Colin_Powell
499	George_W_Bush	George_W_Bush
500	Junichiro_Koizumi	Junichiro_Koizumi

Berikut ini adalah citra wajah (beserta ID-nya) dari 500 hasil prediksi diatas.


25


26


27


28


29


30


31


32


33


34


35


36


37


38


39


40


41


42


43


44


45


46


47


48


49


50


51


52


53


54


55


56


57


58


59


60


61


62


63


64


65


66


67


68


69


70


71


72


73


74


75


76


77


78


79


80


81


82


83


84


85


86


87

88


89


90


91


92


93


94


95


96


97


98


99


100


101


102


103


104


105


106


107


108


109


110


111


112


113


114


115


116


117


118


119


120


121


122


123


124


125


126


127


128


129


130


131


132


133


134


135


136


137


138


139


140


141


142


143


144


145


146


147


148


149


150


151


152


153


154


155


156


157


158


159


160


161


162


163


164


165


166


167


168


169


170


171


172


173


174


175


176


177


178


179


180


181


182


183


184


185


186


187


188


189


190


191


192


193


194


195


196


197


198


199


200


201


202


203


204


205


206


207


208


209


210


211


212


213


214


215


216


217


218


219


220


221


222


223


224


225


226


227


228


229


230


231


232


233


234


235


236


237


238


239


240


241


242


243


244


245


246


247


248


249


250


251


252


253


254


255


256


257


258


259


260


261


262


263


264


265


266


267


268


269


270


271


272


273


274


275


276


277


278


279


280


281


282


283


284


285


286


287


288


289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312


313


314


315


316


317


318


319


320


321


322


323


324


325


326


327


328


329


330


331


332


333


334


335


336


337


338


339


340


341


342


343


344


345


346


347


348


349


350


351


352


353


354


355


356


357


358


359


360


361


362


363


364


365


366


367


368


369


370


371


372


373


374


375


376


377


378


379


380


381


382


383


384


385


386


387


388


389


390


391


392


393


394


395


396


397


398


399


400


401


402


403

404


405


406


407


408


409


410


411


412


413


414


415


416


417


418


419


420


421


422


423


424


425


426


427


428


429


430


431


432


433


434


435


436


437


438


439


440


441


442


443


444


445


446


447


448


449


450


451


452


453


454


455


456


457


458


459


460


461


462


463


464


465


466


467


468


469


470


471


472


473


474


475


476


477


478


479


480


481


482


483


484


485


486


487


488


489


490


491


492


493


494


495


496


497

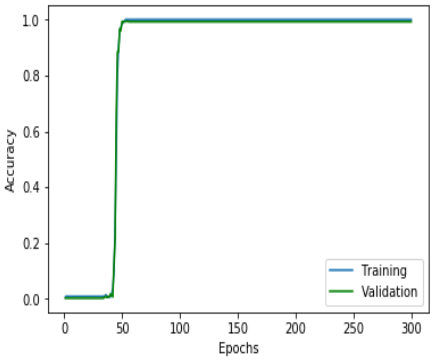

498


499

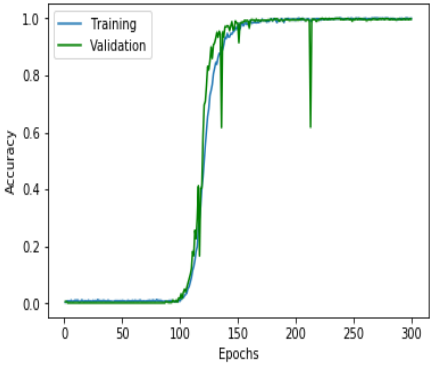

500

Seluruh Hasil Uji Coba

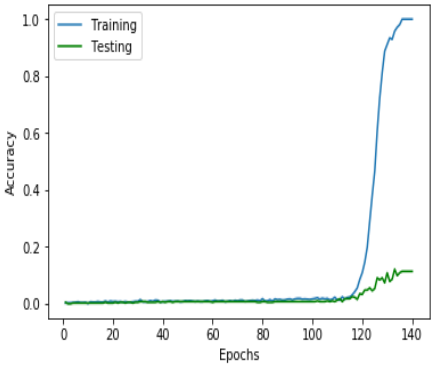
Dataset: Faces94
Activation: LeakyReLU
Dataset Splitting: 70:30
Face align: Yes
Augmentation: No
Batch size: 6



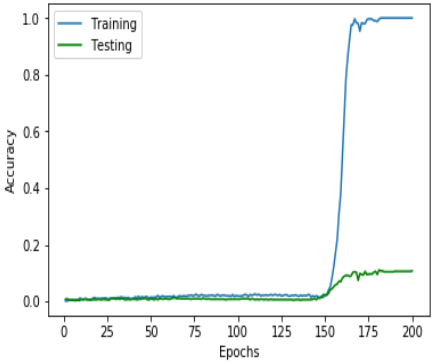
Dataset: Faces94
Activation: ReLU
Dataset Splitting: 70:30
Face Align: Yes
Augmentation: No
Batch size: 6



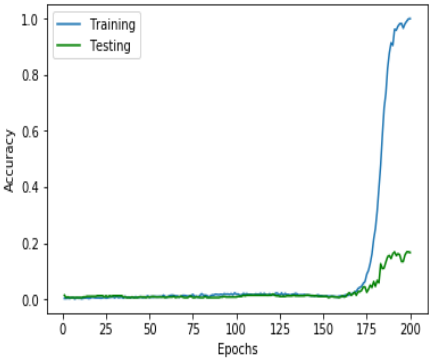
Dataset: LFW (Variasi A), 158 kelas
Total data: 1,580
Activation: LeakyReLU
Dataset Splitting: 70:30
Face align: No
Augmentation: No
Batch size: 6
Testing loss: 10.545
Testing accuracy: 0.135



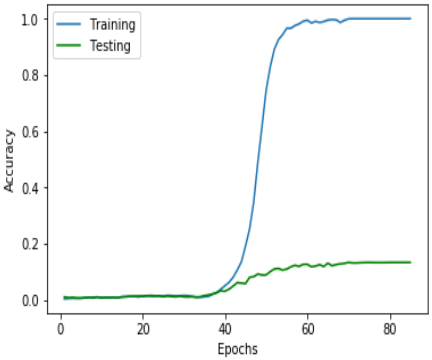
Dataset: LFW (Variasi A), 158 kelas
Total data: 1,580
Activation: LeakyReLU
Dataset Splitting: 60:40
Face align: No
Augmentation: No
Batch size: 6



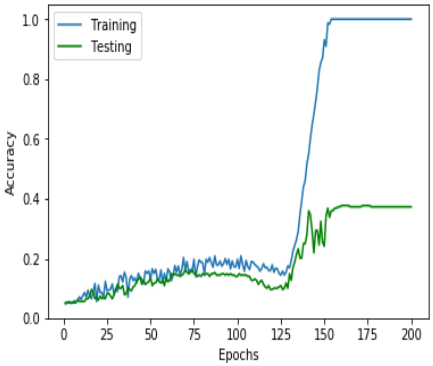
Dataset: LFW (Variasi A), 158 kelas
Total data: 1,580
Activation: LeakyReLU
Dataset splitting: 30
Face align: Yes
Augmentation: No
Batch size: 6
Testing loss: 7.445
Testing accuracy: 0.183



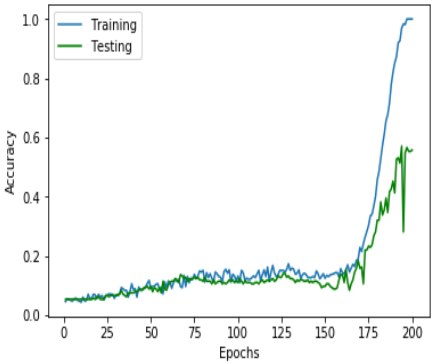
Dataset: LFW (Variasi A), 158 kelas
Total data: 7,900
Activation: LeakyReLU
Dataset splitting: Stratified dataset (70:30)
Face align: No
Augmentation: Transformasi Geometri
Batch size : 6
Testing loss: 10.549
Testing accuracy: 0.132



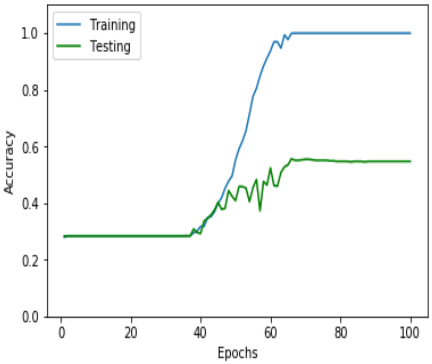
Dataset: LFW (Variasi B), 19 kelas
Total data: 760
Activation: LeakyReLU
Dataset splitting: 70:30
Face align: No
Augmentation: No
Batch size: 6
Testing loss: 4.981
Testing accuracy: 0.372



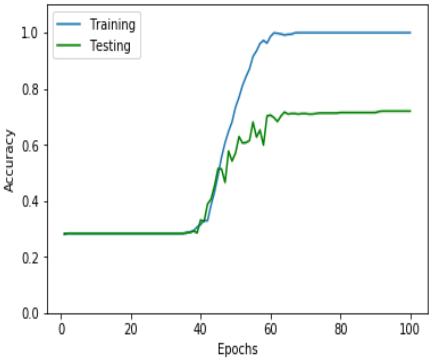
Dataset: LFW (Variasi B), 19 kelas
Total data: 760
Activation: LeakyReLU
Dataset splitting: 70:30
Face align: Yes
Augmentation: No
Batch size: 6
Testing loss: 2.999
Testing accuracy: 0.557



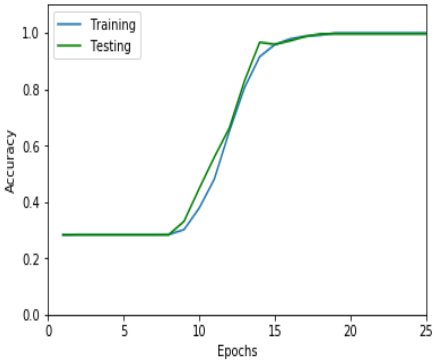
Dataset: LFW (Variasi C), 19 kelas
Total data: 1,867
Activation: LeakyReLU
Dataset splitting: 70:30
Face align: No
Augmentation: No
Batch size: 6
Testing loss: 3.712
Testing accuracy: 0.549



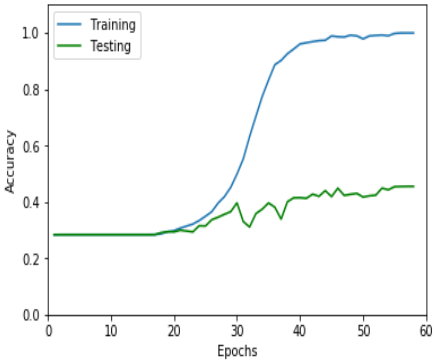
Dataset: LFW (Variasi C), 19 kelas
Total data: 1,867
Activation: LeakyReLU
Dataset splitting: 70:30
Face align: Yes
Augmentation: No
Batch size: 6
Testing loss: 2.365
Testing accuracy: 0.721



Dataset: LFW (Variasi C), 19 kelas
Total data: 9,335
Activation: LeakyReLU
Dataset splitting: 70:30
Face align: No
Augmentation: Filtering
Batch size: 6
Testing loss: 0.028
Testing accuracy: 0.996



Dataset: LFW (Variasi C), 19 kelas
Total data: 9,335
Activation: LeakyReLU
Dataset splitting: 70:30
Face align: No
Augmentation: Transformasi Geometri
Batch size: 6
Testing loss: 4.264
Testing accuracy: 0.455



Dataset: LFW (Variasi C), 19 kelas
Total data: 9,335
Activation: LeakyReLU
Dataset splitting: 70:30
Face align: Yes
Augmentation: Filtering
Batch size: 6
Testing loss: 0.026
Testing accuracy: 0.996